Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Klemen Železnik

Sistem enkratne prijave Shibboleth

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: dr. Andrej Brodnik

Ljubljana, 2015

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge: Sistem enkratne prijave Shibboleth

Najpomembnejše preoblikovanje kateremu smo priča na Internetu je storitvena arhitektura in za njo potrebna infrastruktura. Tako je Internet postal ekosistem kopice storitev, ki jih uporabniki uporabljamo. Za uporabo storitev slednje pogosto želijo uporabnika identificirati, avtenticirati in avtorizirati (IAA). Ker število storitev neprestano raste, tudi uporabniki dobivamo vedno več identitet, kar postaja nepraktično. Kot odgovor na ta problem se je pojavila infrastruktura, ki ponuja možnost enkratne prijave. V diplomski nalogi preučite sisteme za enkratno prijavo in še posebej standard SAML. Na podlagi standarda postavite sistem s pomočjo vmesne plasti Shibboleth ter postavitev ovrednotite.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Klemen Železnik sem avtor diplomskega dela z naslovom:

Sistem enkratne prijave Shibboleth

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Andreja Brodnika,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) in ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela ter
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu prek univerzitetnega spletnega arhiva.

V Ljubljani, dne 18. marec 2015 Podpis avtorja:

Zahvaljujem se profesorju doc. dr. Andreju Brodniku za mentorstvo pri diplomskem delu. Posebna zahvala pa gre tudi moji družini, ki je bila potrpežljiva in mi je stala ob strani v času študija.

Moji družini.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
DS	Discovery Service	storitev odkrivanja
ECP	Enhanced Client or Proxy	okrepljeni klient ali proxy
FQDN	Fully Qualified Domain Name	popolno kvalificirano ime domene
FTP	File Transfer Protocol	protokol za prenos datotek
HDD	Hard Disk Drive	trdi disk
HTTP	HyperText Transfer Protocol	protokol za prenos hiperteksta
HTTPS	HyperText Transfer Protocol Secure	varni protokol za pren os hiperteksta
ID	Identity	identiteta
IDP	Identity Provider	ponudnik identitete
IIS	Internet Information Services	informacijske internetne storitve
IP	Internet Protocol	internetni protokol
ISAPI	Internet Server Application Programming Interface	programski vmesnik intern etnega strežnika
IT	Information Technology	informacijska tehnologija
JRE	Java Runtime Environment	izvajalno okolje Java
LDAP	Lightweight Directory Access Protocol	lahki protokol za dostop do imenika
PEP	Packetized Ensemble Protocol	protokol paketrirane sestave
PKI	Public Key Infrastructure	infrastruktura javnih ključev
SAML	Security Assertion Markup Language	jezik varnostnih trditev

kratica	**angleško**	**slovensko**
SMTP	Simple Mail Transfer Protocol	preprosti protokol za prenos epošte
SOAP	Simple Object Access Protocol	protokol za dostop do preprostih objektov
SP	Service Provider	ponudnik storitve
SSL	Secure Sockets Layer	varna vtična plast
SSO	Single Sign-on	enkratna prijava
TLS	Transport Layer Security	varnost transportne plasti
URI	Uniform Resource Identifier	enotni identifikator vira
URL	Uniform Resource Locator	enolični krajevnik vira
VM	Virtual Machine	virtualna naprava
WAP	Wireless Application Protocol	protokol brezžične aplikacije
WAYF	Where Are You From	od kod prihajaš
XML	Extensible Markup Language	razširljiv označevalni jezik

Povzetek

Cilj protokola Shibboleth je predvsem razbremenitev večkratne avtentikacije uporabnikov znotraj na primer večjih javnih ali zasebnih ustanov, kot so univerze, podjetja, ki imajo razne aplikacije ali storitve, ki potrebujejo avtentikacijo. Protokol Shibboleth deluje na principu enkratne prijave, kar pomeni, da bi uporabnik potreboval le eno uporabniško ime in geslo za prijavo v vse aplikacije znotraj neke ustanove. V delu bomo najprej opisali koncepte protokola Shibboleth in njegove povezave z drugimi mehanizmi predvsem z mehanizmom enkratne prijave (SSO). Poudarek bomo dali na SAML (*Security Assertion Markup Language*), tj. protokol, na katerem se Shibboleth bazira in sklicuje. Opisali bomo delovanje IDP (ponudnik identitete) in SP (ponudnik storitev) ter potek prijave med njimi.

V drugem delu diplomskega dela bomo opisali potek postavitve in konfiguracijo prototipa protokola Shibboleth v lokalnem omrežju, temelječega na operacijskem sistemu Windows.

Ključne besede: Shibboleth protokol, enkratna prijava, SAML, ponudnik storitev, ponudnik identitet.

Abstract

The aim of Shibboleth protocol is mainly to relieve users within, for example, large public or private institutions such as universities, companies that have a variety of different applications or services that require authentication. Shibboleth protocol operates on the principle of single sign-on, which means that the user needs only one username and password to log in to all applications within an institution. In the first part, we will describe the research and concepts of the Shibboleth protocol and its links with other mechanisms, in particular with the mechanism single sign-on. The emphasis will be placed on SAML (*Security Assertion Markup Language*) protocol which Shibboleth is based on and referred to. We will also describe the operation of the IDP (identity provide), SP (service provider) and the steps that take part between them in the registration process.

In the second part of the thesis, we will describe the course of implementation and configuration of the prototype Shibboleth protocol on a local network based on Windows systems.

Keywords: Shibboleth protocol, Single sign-in, SAML, Service provider, Identity provider.

Poglavje 1

Uvod

V današnjem svetu se z razvojem veliko različnih aplikacij, s storitvami in portali računalniški uporabniki soočajo z registracijo na vsakem koraku, kar pripelje do prenasičenosti uporabnika s preveč običajno različnimi uporabniškimi imeni ter gesli, kar velikokrat pripelje do njihove pozabe.

Trenutno je na omrežju več različnih mehanizmov dostopa do podatkov in identitet. Večina jih ponuja standardni dostopni mehanizem do podatkov. Standardni zato, ker če hoče uporabnik dostopati do množice danih aplikacij, mora za vsako vpisati vsakič ponovno svoje uporabniško ime in geslo. Le peščica jih uporablja mehanizem enkratne prijave za dostop do vseh aplikacij. Trenutno eden najboljših mehanizmov enkratne prijave je Shibboleth.

1.1 Shibboleth

Shibboleth, centralna točka tega projekta, je eden najbolj zaželenih in zahtevanih protokolov, ki so uporabljeni v svetovnem spletu ali okoljih, uporabljenih za zaščito sredstev pred nelegitimnimi uporabniki. Ena največjih prednosti protokola Shibboleth je implementacija enkratne prijave skozi visokonivojski protokol, imenovan SAML (*Security Assertion Markup Language*). Shibboleth postaja zelo popularen in zaželen po fakultetah, univerzah, bankah, tehnoloških centrih ter ostalih krajih, in sicer predvsem, ker je od-

prtokodna, prožna ter zelo močna infrastrukturna platforma. [2]

Protokol je dobil ime Shibboleth po dogodku iz Biblije, ki se je dogajal med dvema semitskima plemenoma – Ephraimiti ter Gileaditi –, med katerima se je dogajala vojna. Gileaditi so premagali Ephramite in postavili blokado, da bi ujeli bežeče Ephramite. Geliaditina straža je od vsake osebe, ki je šla mimo blokade, zahtevala, da izgovori besedo »Shibboleth«. Ephramiti, ki niso znali izgovoriti »š«, so bili tako razkrinkani kot sovražniki in so bili umorjeni. Tako je oseba, ki krši Shibboleth, hitro razkrinkana kot tujec in instantno izločena iz skupine. V angleškem jeziku beseda Shibboleth pomeni test, ki razlikuje eno skupino od druge. [1]

1.2 Struktura naloge

V prvem delu diplomskega dela bomo opisali koncepte protokola Shibboleth in njegove povezave z drugimi mehanizmi, natančneje bomo v drugem poglavju pregledali SSO-mehanizem. Opisali bomo njegovo arhitekturo, postopek prijave in protokole, ki jih uporablja za delovanje. V tretjem poglavju bomo predelali SAML-protokol, ki je protokol, ki implementira možnost enkratne prijave. Najprej bomo opisali njegovo osnovno delovanje in se nato bolj poglobili v samo arhitekturo ter komponente, ki ga sestavljajo. V četrtem poglavju bomo spoznali Shibboleth, ki deluje kot okvir SAML-protokola za implementacijo enkratne prijave. Opisali bomo njegovo arhitekturo in komponente, med katerimi so pomembnejši ponudnik storitve, ponudnik identitete in storitev odkrivanja. Po osnovnem opisu bomo pregledali še postopek korakov in druge elemente ter funkcije, ki pripeljejo do enkratne prijave z implementacijo Shibboleth. V drugem delu diplomskega dela bomo opisali postopek implementacije protokola Shibboleth, ki pripelje do tega, da ko uporabnik dostopi do varovanega vira, ga ta preusmeri na avtentikacijsko stran, kjer mora podati svoje poverilnice, da pride do želenega vira.

Poglavje 2

SSO-mehanizem

2.1 Pregled

Veliko spletnih aplikacij v istem sistemu ali isti organizaciji še vedno zahteva, da se uporabnik registrira vsakič, ko želi uporabiti individualno aplikacijo. Širjenje spletnih aplikacij in storitev je pripeljala do tega, da to postaja nepraktično in nepotrebno, da si morajo uporabniki zapomniti vsa uporabniška imena ter gesla za vse aplikacije, do katerih želijo dostopati. Protokol enkratne prijave (SSO) omogoča uporabnikom, da potrebujejo samo eno uporabniško ime in geslo za dostopanje do več različnih aplikacij ter storitev. Prednosti, ki jih enkratna prijava ponuja uporabnikom, so:

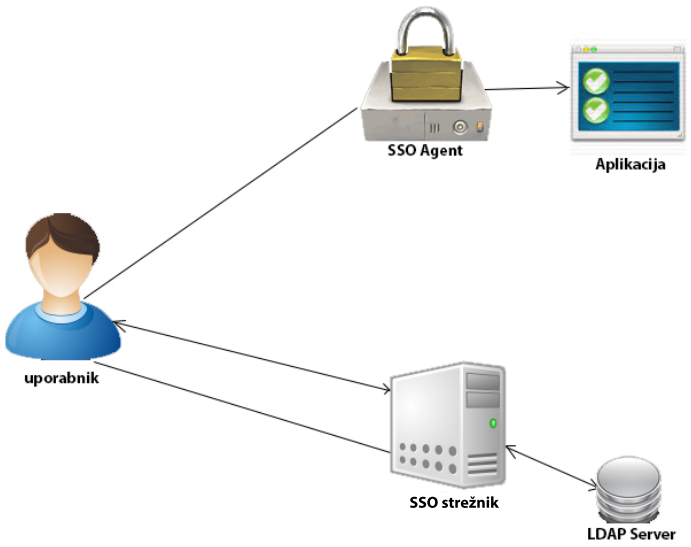
- Zmanjšanje zmožnosti pozabe uporabniškega imena in gesla zaradi prenasičenosti teh.

- Zmanjšanje porabe časa vnašanja različnih uporabniških imen in gesel za različne aplikacije ter storitve. Vedno se vnašata isto uporabniško ime in geslo, pri čemer prihranimo čas, saj redkeje zgrešimo geslo, kar pripelje do njegovega ponovnega vpisovanja.

- Zmanjšanje stroškov IT zaradi zmanjševanja klicev uporabnikov zaradi pozabljenih gesel.

2.2 SSO-arhitektura

V SSO-mehanizmu obstajajo štirje glavni akterji, ki delujejo med seboj: uporabnik, avtentikacijski strežnik (SSO-strežnik), avtentikacijski agent (SSO-agent) in sama aplikacija, v katero se dostopa. Slika 2.1 prikazuje njihovo medsebojno arhitekturo.



Slika 2.1: Poenostavljena Web SSO arhitektura

SSO-strežnik je centralna entiteta v SSO-sistemu, saj shranjuje centralizirane poverilnice in skrbi za avtentikacijo uporabnika, vzpostavitev povezave med uporabnikom ter drugimi elementi in širjenje identitete uporabnika med aplikacijami.

Postopek se začne, ko uporabnik posreduje svoje uporabniško ime in geslo SSO-strežniku. Strežnik nato preveri parametre, ki jih je dobil, in avtenticira uporabnika, če se parametri ujemajo s kakšnimi shranjenimi poverilnicami ali certifikati. Ko je uporabnik avtenticiran, strežnik ohranjuje uporabnikovo sejo s tem, da nastavi HTTP-piškotek ali žeton uporabniku. Podatki iz piškotkov so zavarovani in se potrebujejo za identifikacijo uporabnika v nadaljnjih dostopih do strežnika.

Uporabniški agent je običajno integriran v sami ciljni aplikaciji kot neka knjižnica, apache ali IIS-modul. Uporabniški agent preveri, ali je uporabnik dejansko overjen za dostop do aplikacije. Če ta ni overjen, ga agent pošlje nazaj na SSO-strežnik. Če je bil uporabnik avtenticiran, agent še enkrat preveri vir podatkov in jih prenese na ciljno aplikacijo ter poveže uporabnika z aplikacijo.

2.3 Postopek prijave SSO

1. korak: Uporabnik dostopa do vira
Uporabnik začne s poskusom dostopa do varovanega vira. Monitor vira ugotovi, ali ima uporabnik aktivno sejo in če je nima, ga usmerja k ponudniku storitev, da bi začeli postopek SSO.

2. korak: Ponudnik izda zahtevo overovitve
Uporabnik prispe do ponudnika storitev, ki pripravi zahtevo za preverjanje pristnosti in pošlje zahtevo ter uporabnika do ponudnika identitete. Programska oprema ponudnika storitve je običajno nameščena na istem strežniku kot vir.

3. korak: Uporabnik je avtenticiran pri ponudniku identitete
Ko uporabnik prispe do ponudnika identitete, ta preveri, ali ima uporabnik že obstoječo sejo. Če je tako, nadaljuje na naslednji korak. Če ni tako, pa uporabnika ponudnik identitete avtenticira s tem, da ga na primer prosi za uporabniško ime ter geslo. Ko to stori, gre uporabnik na naslednji korak.

4. korak: Ponudnik identitete izda odziv preverjanja pristnosti
Po identifikaciji uporabnika ponudnik identitete pripravi odziv avtentikacije in pošlje tega ter uporabnika nazaj do ponudnika storitve.

5. korak: Ponudnik storitve preveri odziv avtentikacije
Ko uporabnik prispe z odgovorom ponudnika identitete, ponudnik storitve potrdi odgovor, ustvari sejo za uporabnika in nekaj podatkov, prejetih z odzivom, na primer uporabnikov ID, ki je na voljo varovanemu viru. Po tem je uporabnik poslan na dostopani vir.

6. korak: Vir vrne vsebino
Kot v 1. koraku uporabnik spet poskuša dostopati do varovanega vira, a tokrat ima uporabnik sejo in vir, ve, kdo poskuša dostopati do njega. S temi informacijami bo vir servisiral uporabnikovo zahtevo in poslal nazaj zahtevane podatke.

2.4 SSO-protokoli

V tem delu bomo na kratko opisali nekaj glavnih protokolov, ki implementirajo princip enkratne prijave.

- **Kerberos** protokol je ustvaril MIT kot rešitev za probleme, ki nastajajo zaradi varnosti omrežja. Protokol Kerberos uporablja močno šifriranje, tako da stranka lahko dokaže svojo identiteto strežniku čez nevarovano omrežje. Ko odjemalec in strežnik uporabita kerberos, da dokažejo svojo identiteto, lahko tudi šifrirata vse njihove komunikacije za zagotovitev zasebnosti in celovitosti podatkov, ki jih uporabljajo v svoje poslovanju. [17]

- **Integrated Windows Authentication**, poznan tudi kot NTLM, pri preverjanju pristnosti zgoščuje uporabniška imena in gesla, preden jih pošlje prek omrežja. [11]

- **OpenID SSO**: identiteta se deli glede na njihovo domeno. Ko uporabnik poskuša dostopati do deljene vsebine ali aplikacije vnese svoj openeID, ki mu je bil dodeljen glede na domeno ponudnika OpenID, kot so Google, Yahoo, flick itd. [12]

- **SAML(*Secure Assertion Marku-up Language*)** je spletni varnostni protokol, ki temelji na XML- (Extended Mark-up Language) arhitekturi in implementira veliko varnostnik mehanizmov znotraj spletnih domen prek interneta. Eden ključnih vidikov protokola je uporabniška avtentikacija in avtorizacija prek enkratne prijave. [4] Ta protokol je

tudi uporabljen v tem diplomskem delu, saj je del implementacije Shibboletha, in bo opisan podrobneje v naslednjem poglavju.

Poglavje 3

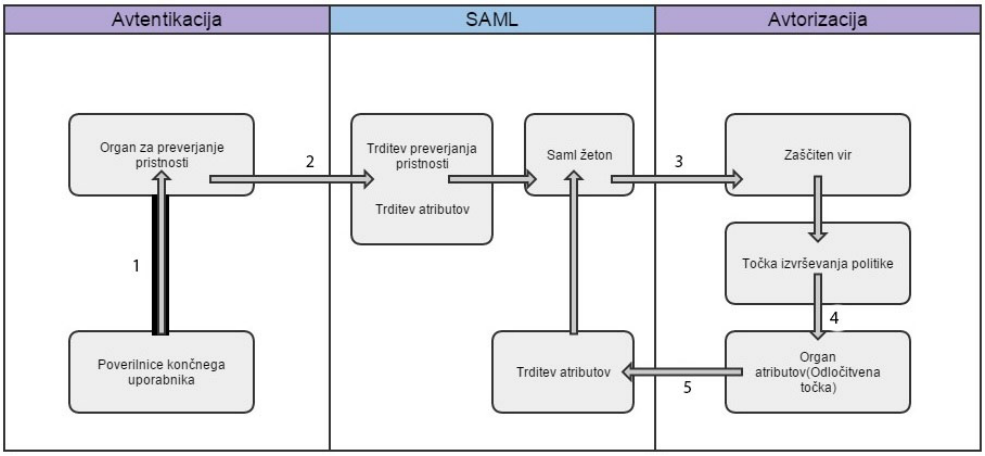
Protokol SAML

SAML (*Security Assertion Mark-up Language*) temelji na XML-formatu za izmenjavo podatkov za preverjanje pristnosti in dovoljenj med strankami, zlasti med ponudnikom identitet ter ponudnikom storitve. Gre za zamisel, da lahko eno spletno mesto jamči za identiteto in atribute uporabnika enemu ali več drugim spletnim mestom. [4] Dva glavna elementa sta ponudnik identitete (IDP- *Identity Provider*) in ponudnik storitev (SP-*Service Provider*). Pri tem je kritično, da potrdila o identiteti in atributih uporabnikov niso dostopna nepooblaščenim, zato vse kritične podatke šifriramo in elektronsko podpišemo. SAML je izdelek OASIS Security Services Committeja. SAML sega vse od leta 2001. Najnovejša večja posodobitev je bila objavljena leta 2005, vendar so protokol izboljševali redno prek drugih dodatnih neobveznih standardov. Ena najpomembnejših lastnosti, ki jo SAML vsebuje, je tudi enkratna prijava.

SAML ne cilja na specifikacijo novih kriptografskih tehnik ali varnostnik modelov, ampak se raje osredotoča in opisuje standardne varnostne tehnologije v industriji prek XML-formata ter se zanaša na že obveljavljene omrežne protokole, kot so HTTP, HTTPS, SSL. V tem delu bomo obravnavali tehnične lastnosti SAML-ja, osredotočeno na najnovejšo verzijo SAML 2.0, ki je ključnega pomena za razumevanje Shibboleth, ki je ena od njegovih implementacij.

3.1 Osnovno delovanje SAML-protokola

Uporaba SAML-protokola je različna, in sicer glede na primer uporabe. Ustanove, ki protokol uporabljajo, definirajo svoj primer uporabe, ki temelji na njihovih potrebah. Najbolj pogost primer uporabe je, da uporabnik želi dostopati do varovane aplikacije ali storitve.



Slika 3.1: Primer osnovnega poteka uporabe SAML.

V tem primeru (slika 3.1) je predvideno, da bo uporabnik dostopal prek tipičnega brskalnika do varovane vsebine, ki je spletna aplikacija.

1. Uporabnik preda svojo poverilnico, ki je ponavadi uporabniško ime in geslo, organu za preverjanja pristnosti, ki je lahko katera koli varnostna ali avtorizacijska aplikacija, ki podpira SAML.

2. Organ za preverjanja pristnosti potrdi uporabnikove dostopne parametre in sproducira zagotovilo preverjanja pristnosti skupaj z enim ali več trditvami atributov, ki so lahko katera koli informacija o uporabnikovem profilu, kot so njegova funkcija, polno ime ali naslov elektronske pošte. Uporabnik je nato lahko avtenticiran in identificiran s SAML-trditvami ter žetonom.

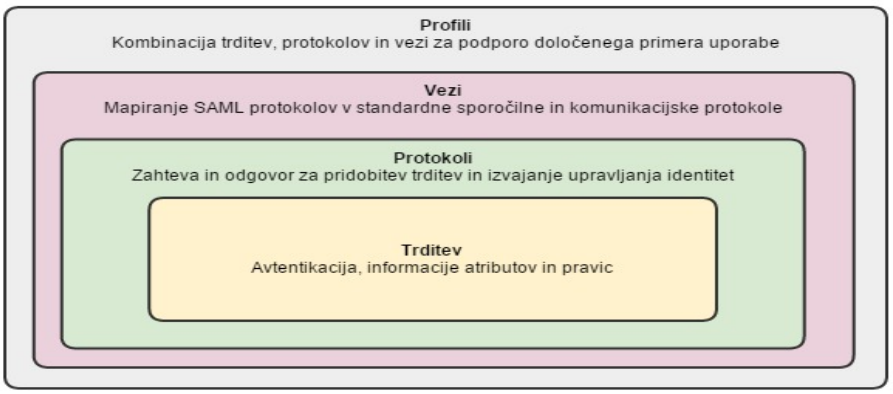
3. Uporabnik poskuša pridobiti dostop do vira z uporabo SAML-žetona.

4. PEP (točka izvrševanja politike) takoj prestreže uporabnikovo zahtevo do vira in nato posreduje uporabnikov žeton ter trditev preverjanja pristnosti organu atributov, ki je lahko vsaka varnostna aplikacija, ki podpira SAML.

5. Glede na svojo politiko organ atributov odloči, ali uporabnik lahko dostopa do vira ali ne. Če lahko, potem je generirana trditev atributov in vezana na uporabnikov žeton. Zdaj je lahko SAML-žeton predan zaupanja vrednim partnerjem, ki so nabrani v tem odnosu enkratne prijave.

3.2 SAML-arhitektura

V tem delu bomo opisali strukturo in glavne komponente, ki sestavljajo SAML-protokol in ki prispevajo k vzpostavljanju vzdrževanja ter izdajanju varnostnih izmenjav podatkov znotraj zaupnih organizacij. Te komponente tipično dovoljujejo izmenjavo avtentikacije, avtorizacije, atributov in tako dalje prek skupnih virov. [4]



Slika 3.2: SAML-struktura.

Podatki SAML in trditve so kodirani v obliki sheme XML, vključno z

osnovnimi informacijami, ki opredeljujejo in določajo identifikator, ki se uporablja za datum trditve, čas in ime izdaje ter časovno režo ali interval, za katerega obstaja validacija trditve. Predložitev SAML-trditev do avtorizacijskih in avtentikacijskih odločitvenih točk poteka prek protokolov zahtev in odgovorov. Slika 3.2 prikazuje osnovno strukturo SAML-protokola.
Dva druga pomembna koncepta SAML-namestitve:

- **Metapodatki** dovolijo SAML-strankam, da komunicirajo med seboj in si izmenjujejo podatke. Stranke se prepoznajo med seboj skozi metapodatke. Na primer: ponudnik storitve (SP) pozna profil ponudnika identitete (IDP) skozi njegove metapodatke in obratno. SAML-metapodatki so specificirani in definirani po svoji pravilni vsebini XML-datoteke. Shibboleth v trenutnem stanju ne ponuja orodja za izvoz ali uvoz SAML-metapodatkov. Raje uporablja XML posredno kot mehanizem za konfiguracijo, ki enumerira niz zaupanja vrednih partnerjev in pove programski opremi, kako varno komunicirati z njimi. [9] Ponudnik identitete porabi metapodatke, tako da išče entitete, ki se izdajajo za ponudnike storitev. Nasprotno pa ponudnik storitve uporablja metapodatke, tako da išče entite, ki se izdajajo za ponudnika identitete. Vsak ponudnik potrebuje za pravilno delovanje metapodatke svojega nasprotnika. To pomeni: če med avtentikacijo ponudnik storitve ne dobi metapodatkov ponudnika identitete, in obratno, ne moremo zagotoviti, da smo prispeli na pravilno mesto.

- SAML **kontekst za preverjanje pristnosti** je url, ki določa načine overjanja v avtentikacijskih zahtevkih in izjavah preverjanja pristnosti SAML. Uporablja se za prenašanje informacij v zvezi z močjo avtentikacije, ki jo je uporabnik izkoristil, ko se je avtenticiral.

3.3 SAML-komponente

SAML-strukturo in okolje sestavlja več komponent, ki jih bomo natančneje opisali ter podali nekaj konkretnih primerov uporabe.

3.3.1 Trditve

SAML dovoljuje eni stranki, da jamči, da so atributi subjekta pravilni in verodostojni. Na primer: SAML-trditev trdi, da ima uporabnik Miha Muha administrativne pravice, njegove e-mail je miha.muha@mail.si in je uporabnik skupine inženirjev. Ključna elementa v trditvi sta subjekt in vsebina izjave. SAML definira tri izjave, ki so lahko prenesene s trditvijo. [4] **Izjave za preverjanje pristnosti** so izdane s strani stranke, ki je uspešno avtenticirala uporabnika. Definirajo, kdo je izdal izjavo, način prijave, obdobje veljavnosti in druge informacije, povezane s preverjanjem pristnosti. Spodnji primer (slika 3.3) prikazuje preproste SAML-izjave za preverjanje pristnosti. Za ta primer trditev ne vsebuje neobveznih SAML-podatkov niti ne deklarira zahtevanega SAML-imenskega prostora. Izjava navaja, da se je uporabnik cn=joe, o=novell avtenticiral ob specifičnem času z uporabljeno metodo prijave, ki uporablja geslo.

```
<saml:Assertion MajorVersion="1"
            MinorVersion="0"
            AssertionID="some_unique_identifier"
            Issuer="novell_issuer_id"
            IssueInstant="DATE_TIME"
<saml:AuthenticationStatement
      AuthenticationMethod="urn:oasis:names:tc:SAML: 1.0: am:password"
      AuthenticationInstant="DATE_TIME">
   <saml:Subject>
         <saml:NameIdentifier>cn=joe,o=novell</saml:NameIdentifier>
   </saml:Subject>
  </saml:AthenticationStatement>
</saml:Assertion>
```

Slika 3.3: Primer izjave za preverjanje pristnosti.

Izjave atributov trdijo, da ima subjekt A atribut S v imenskem prostoru N z vrednostmi V. Stranka se lahko poveže s subjektom s pridobljenimi atributi. To dovoljuje stranki, da kreira in prilagodi svoje aplikacije za zunanje uporabnike. Prav tako omogoča uporabnikom, ki dostopajo do spletnega mesta z uporabo SAML, da so preskrbljeni

z ustreznimi uporabniškimi podatki. Spodnji primer (slika 3.4) prikazuje preprosto SAML-izjavo atributov. Za ta primer izjava ne vsebuje neobveznih SAML-podatkov niti ne deklarira zahtevanega SAML-imenskega prostora. Izjava navaja, da ima uporabnik cn=joe,o=novell atribut Email v imenskem prostoru urn:test:attributes z vrednostjo joe@novell.com.

```
<saml:Assertion MajorVersion="1"
            MinorVersion="0"
            AssertionID="some_unique_identifier"
            Issuer="the_issuer_identifier"
            IssueInstant="DATE_TIME"
  <saml:AttributeStatement
    <saml:Subject>
          <saml:NameIdentifier>cn=joe,o=novell</saml:NameIdentifier>
    <saml:Subject>
    <saml:Attribute AttributeName="Email"
              AttributeNamespace="urn:test:attributes">
    </saml:AttributeValue>joe@novell.com</saml:AttributeValue>
    <saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>
```

Slika 3.4: Primer izjave atributa.

Izjave odločitev o izdaji dovoljenja izda PDP (*Policy Decision Point*) in se uporabljajo za določitev, ali se subjektu S dovoli dostop do R vira, ki je vir, do katerega želi subjekt dostopati z vrsto dostopa A glede na pridobljene dokaze. Subjekt je lahko katera koli entiteta, naj bo to človek ali program, ki želi dostopati do vira. Vir lahko predstavlja katere koli podatke ali storitve, ki zahtevajo nadzor dostopa. Primer (slika 3.5) prikazuje preprosto trditev izjave odločitev o izdaji dovoljenja. Izjava navaja, da ima uporabnik cn=joe,o=novell dovoljenje za dostop do Buy opcije na viru `uri:novell.com/buy_stuff`.

```
<saml:Assertion MajorVersion="1"
            MinorVersion="0î
            AssertionID="some_unique_identifier"
            Issuer="the_issuer_identifier"
            IssueInstant="DATE_TIME"
  <saml:AuthorizationDecisionStatement
     Resource="uri:novell.com/buy_stuff"
     Decision="saml:Grant">
    <saml:Subject>
          <saml:NameIdentifier>cn=joe,o=novell</saml:NameIdentifier>
    </saml:Subject>
    <saml:Action>Buy</saml:Action>
  </saml:AuthorizationDecisionStatement>
</saml:Assertion>
```

Slika 3.5: Primer izjave odločitev o izdaji dovoljenja.

3.3.2 Protokoli

SAML opredeljuje številne zahteve/odzive protokolov. Protokol je kodiran v XML-shemi kot niz parov zahtev in odzivov. Definirani protokoli so: [4]

```
<samlp:AuthnRequest
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  ID="aaf23196-1773-2113-474a-fe114412ab72"
  Version="2.0"
  IssueInstant="2004-12-05T09:21:59"
  AssertionConsumerServiceIndex="0"
  AttributeConsumingServiceIndex="0">
  <saml:Issuer>https://sp.example.com/SAML2</saml:Issuer>
  <samlp:NameIDPolicy
    AllowCreate="true"
    Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"/>
</samlp:AuthnRequest>
```

Slika 3.6: Primer izjave odločitev o izdaji dovoljenja.

Protokol za zahtevo avtentikacije: definira `<AuthnRequest>` sporočilo (slika 3.6), ki povzroči, da se vrne `<Response>` odgovor, ki vsebuje eno ali več trditev, ki se nanašajo na zavezanca. Tipično je `<AuthnRequest>` izdan s strani ponudnika storitve, medtem ko ponudnik identeitete vrne

`<Response>` sporočilo. Uporablja se za podporo profilov enkratne prijave (Web browser SSO). `<AuthnRequest>` (slika 3.6), ki implicitno zahteva trditev, ki vsebuje izjavo za preverjanje pristnosti, je bila izdana s strani ponudnika storitve in nato predstavljena ponudniku identitete. Ponudnik identitete nato avtorizira glavnico in izda odziv, ki je poslan nazaj do ponudnika storitve.

Artefakt protokol: SAML-sporočilo se prenaša z ene entitete na drugo, bodisi po vrednosti bodisi z referenco. Sklicevanje na SAML-sporočilo se imenuje artefakt. Sprejemnik artefakta rešuje sklicevanja s pošiljanjem `<samlp:ArtifactResolve>` zahtev direktno na izdajatelja artefakta, ki nato reagira z dejanskim sporočilom, ki ga je sklical artefakt. Vzemimo za primer, da ponudnik identitete pošlje naslednjo `<samlp:ArtifactResolve>` zahtevo (slika 3.7) neposredno na ponudnika storitve.

```
<samlp:ArtifactResolve
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  ID="_cce4ee769ed970b501d680f697989d14"
  Version="2.0"
  IssueInstant="2004-12-05T09:21:58">
  <saml:Issuer>https://idp.example.org/SAML2</saml:Issuer>
  <!-- an ArtifactResolve message SHOULD be signed -->
  <ds:Signature
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#">...</ds:Signature>
  <samlp:Artifact>AAQAAMh48/1oXIM+sDo7Dh2qMp1HM4IF5DaRNmDj6RdUmllwn9jJHyEgIi8=</samlp:Artifact>
</samlp:ArtifactResolve>
```

Slika 3.7: Primer artefakt protokola.

V odgovor ponudnik storitev vrne SAML-element, sklicujoč se na priloženi artefakt. Ta protokol je sestavna podlaga vezave HTTP artefakt.

Protokol za menedžment identifikacije imena: zagotavlja mehanizem za spremembo vrednosti ali obliko imena glavnice. Izdajatelj tega ukaza je lahko ponudnik identitete (IDP) ali ponudnik storitve (SP). Protokol ponuja tudi mehanizem za prekinitev združenja imena med ponudnikom identitete (IDP) in ponudnikom storitve (SP). V scenariju je ponudnik identitete izmenjal nekaj identifikatorjev za glavnico

s ponudnikom storitev, ki jim omogoča, da delijo skupen identifikator za nekaj časa. Pozneje lahko ponudnik identitete obvesti ponudnika storitve o zamenjavi v formatu ali vrednost, ki jo bo uporabljal za identifikacijo iste glavnice v prihodnosti. Možno je tudi, da eden od ponudnikov želi obvestiti drugega, da ne bo več izdajal ali sprejemal sporočila z uporabo določenega identifikatorja. Protokol za menedžment identifikacije se uporablja za implementacijo teh scenarijev.

Protokol enkratne odjave: definira zahtevo, ki dovoli uporabniku skoraj hkratno odjavo iz vseh sej, povezanih z glavnico. Odjava se lahko zgodi ročno ali pa se avtomatsko sproži zaradi poteka časa seje. `<samlp:LogoutRequest>` določa, kdo se bo odjavil z elementom NameID. V našem primeru (slika 3.8) je to Alice in jo prepoznamo prek elektronskega naslova.

```
<?xml version="1.0"?>
<samlp:LogoutRequest Destination="https://idp.acme.biz" ID="logoutrequest1">
  <saml:NameID>alice@acme.biz</saml:NameID>
</samlp:LogoutRequest>
```

Slika 3.8: Zahteva protokola enotne odjave.

Lahko bi jo prepoznali tudi prek uporabniškega imena ali katere druge identifikacije. Ko enkrat ponudnik identitete pridobi `<LogoutRequest>`, bo kreiral `<LogoutRequest>`, ki ga pošlje na druge seje, v tem primeru v aplikacijo Cars. Če je `<LogoutRequest>` veljaven, bo Cars prekinil sejo Alice in vrnil ponudniku identitete nazaj `<LogoutResponse>` z vsebino, ki potrjuje, da je bila odjava uspešna.

Protokol mapiranja identifikacije imena: zagotavlja mehanizem, ki omogoča povezovanje in preslikavo identifikatorjev v drugega identifikatorja za istega naročnika. Protokol deluje tako, da najprej nekdo zahteva `<NameIDMappingRequest>` sporočilo ponudniku identitete. Nato ponudnik identitete vrne `<NameIDMappingResponse>` sporočilo naročniku.

3.3.3 Vezi

Podrobneje povejo, kako se SAML-protokoli mapirajo na transportne protokole. Na primer: specifikacija SAML omogoča vezavo, ki pove, kako se prenaša SAML-zahteva/odziv s SOAP-izmenjavo sporočil. Definirane vezi so: [4] [15]

Vezava SAML SOAP: definira, kako so SAML-protokolna sporočila prenesena znotraj sporočil SOAP 1.1. Dodatno tudi definira, kako so SOAP-sporočila poslana prek HTTPja. Spodaj (slika 3.9) je primer poizvedbe, ki prosi za SAML organ atributov za trditev, ki vsebuje izjavo atributa.

```
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Body>
        <samlp:AttributeQuery xmlns:samlp:="…"
xmlns:saml="…" xmlns:ds="…" ID="_6c3a4f8b9c2d" Version="2.0"
IssueInstant="2004-03-27T08:41:00Z"
            <ds:Signature> … </ds:Signature>
            <saml:Subject>
            …
            </saml:Subject>
        </samlp:AttributeQuery>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Slika 3.9: Primer vezave SAML SOAP.

Povratne SOAP-vezi (PAOS): definirajo večstopenjsko SOAP-/HTTP-izmenjavo sporočil, ki dovoljujejo HTTP-klientu, da postane SOAP-odzivnik. Uporabljen je v ECP (*Enhanced Client and Proxy Profile*) in zlasti zasnovan za podporo WAP-prehodov.

Deluje tako, da najprej HTTP-naročnik pošlje HTTP-zahtevo do SAML-zahtevnika. SAML-zahtevnik se odzove s HTTP-odzivom, ki vsebuje SOAP-ovojnico, ki vsebuje SAML-sporočilo. Nato HTTP-naročnik pošlje HTTP-zahtevo, ki vsebuje SOAP-ovojnico, ki vsebuje SAML-odgovor, HTTP izvirnemu SAML-zahtevniku. SAML-zahtevnik nato odgovori s HTTP-odzivom, po možnosti v odgovor na prvotno zahtevano storitev.

Preusmerjena HTTP-vezava: definira, kako je lahko SAML-protokolno sporočilo preneseno z uporabo preusmerjenega HTTP-sporočila. Namenjena je za primere, v katerih morata SAML-zahtevnik in odzivnik komunicirati s pomočjo HTTP-uporabniškega agenta kot s posrednikom. Na primer: če stranke ne delijo neposredne poti do komunikacije.

Deluje tako, da najprej uporabnik naredi HTTP-zahtevo na sistem. V teku obdelave zahtevka se sistem odloči za iniciacijo izmenjave SAML-protokola. Sistem, ki deluje kot SAML-zahtevenik, odgovori na HTTP-zahtevo uporabnika iz prejšnjega koraka, s tem da vrne SAML-zahtevo. Ta SAML-zahteva je kodirana v glavi HTTP-odziva. SAML-odzivnik se odzove na zahtevo SAML, s tem da takoj vrne SAML-odgovor. Ko SAML-odgovor prispe, SAML-zahtevnik vrne HTTP-odziv uporabniku.

Vezava HTTP POST: definira, kako je lahko SAML-protokolno sporočilo preneseno znotraj base64 kodirane vsebine HTTP-obrazca. Namenjena je za primere, v katerih morata SAML-zahtevnik in odzivnik komunicirati s pomočjo HTTP-uporabniškega agenta kot s posrednikom. Na primer: če stranke ne delijo neposredne poti do komunikacije. Uporabnik najprej sistemu izda HTTP-zahtevo. V teku obdelave zahteve se sistem odloči za začetek izmenjave SAML-protokola.

Sistem, ki deluje kot SAML-zahtevnik, se odzove na HTTP-zahtevo uporabnika, s tem da vrne SAML-zahtevo. Zahteva je vrnjena v XHTML-dokumentu, ki vsebuje obrazec in vsebino. Uporabnik dostavi SAML-zahtevo z izdajo zahteve HTTP POST SAML-odzivniku. Nato odzivnik vrne SAML-odgovor uporabniku, da ga dostavi SAML-zahtevniku. SAML-odgovor je dostavljen na isti način, kot smo ga prej opisali. Ko SAML-zahtevnik dobi SAML-odgovor, vrne HTTP-odgovor uporabniku.

Vezava HTTP artefakt: definira, kako je prenesena referenca na SAML-odziv/zahtevo prek HTTP-ja. Kot na primer prek poizvedbe

v URL-naslovu. Namenjena je za primere, v katerih morata SAML-zahtevnik in odzivnik komunicirati z uporabo HTTP-uporabniškega agenta kot s posrednikom.

SAML-odzivnik določi SAML-zahtevnik tako, da preuči artefakt postopek in izda SAML-prosilcu, ki uporablja neposredno SAML-vez, `<samlp: ArtifactReslove>` zahtevek, ki vsebuje arteafak. Ob predpostavki, da so izpolnjeni vsi potrebni pogoji, SAML-zahtevnik vrne `<samlp: ArtifactResponse>` (slika 3.10), ki vsebuje prvotno sporočilo SAML-zahtevka, za katerega želi, da ga SAML-odzivnik sprocesira. Odzivnik vrne SAML-artefakt uporabniškemu agentu, da ga ta vrne SAML-zahtevniku. Vrne ga po istem principu kot prej. SAML-zahtevnik določi SAML-odziv s preučevanjem artefakta in nato ponovno kot prej izda `<samlp:ArtifactResolve>` zahtevo, ki vsebuje artefakt, SAML-odzivniku z uporabo direktne SAML-vezave. Ob predpostavki, da so izpolnjeni pogoji, SAML-odzivnik vrne `<samlp: ArtifactResponse>`, ki vsebuje SAML-odzivno sporočilo, ki ga želi sprocesirati. Ob pridobitvi SAML-odziva SAML-zahtevnik vrne HTTP-odgovor uporabniškemu agentu.

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
    <se:Header/>
    <se:Body>
        <sp:ArtifactResolve xmlns:sp="urn:oasis:names:tc:SAML:2.0:protocol" ID="I
                            IssueInstant="2007-01-02T20:48:35Z" Version="2.0">
            <sa:Issuer xmlns:sa="urn:oasis:names:tc:SAML:2.0:assertion">
                http://cgi.cohos.de:80/cgi-bin/zxid?o=B
            </sa:Issuer>
            <sp:Artifact>
                AAQAAAs0WdStWqaEgH6NzkS3McLikBA0INwfqVfFKj9X2iqopCro+DS1MDE=
            </sp:Artifact>
        </sp:ArtifactResolve>
    </se:Body>
</se:Envelope>
```

Slika 3.10: Primer odgovora artefakt.

3.3.4 Profili

Jedro specifikacije SAML definira, kako se SAML-zahteve in odzivi prenašajo. Vseeno je bilo razvitih število primerov uporabe, ki potrebujejo formulacijo profilov, ki definirajo, kako so povezane med seboj SAML-trditve, protokoli in vezi. Nekateri teh profilov so: [4]

Web Browser SSO-profil: definira, kako brskalnik podpira enotno prijavo (SSO), ko uporablja `<AuthnResponse>` protokolna sporočila v uporabi s HTTP Redirect, HTTP Post in HTTP artefakt vezavami.

Deluje tako, da najprej glavnica prek HTTP-uporabniškega agenta izvrši HTTP-zahtevo za varovan vir na ponudniku storitve. Ponudniki storitve nato pridobi lokacijo končne točke na ponudniku identitete za protokol avtentikacijske zahteve, ki podpira priljubljeno vezavo. Sredstva, s katerimi se to izvrši, so odvisna od implementacije. Ponudnik storitve lahko na primer uporabi SAML-profil za odkrivanje identitete. V tretjem koraku ponudnik storitve izda `<AuthnRequest>` sporočilo, ki bo poslano ponudniku identitete s strani uporabniškega agenta. Za prenos sporočila do ponudnika identitete prek uporabniškega agenta lahko uporabimo povratno vezavo HTTP, HTTP POST ali HTTP artefakt. V četrtem koraku je glavnina identificirana s strani ponudnika identitete. Za to je potrebna ponovna avtentikacija ali pa uporaba obstoječe avtentikacijske seje. V petem koraku ponudnik identitete izda `<Response>` sporočilo, ki mora biti dostavljeno ponudniku storitve prek uporabniškega agenta. V zadnjem koraku, ko smo dobili odgovor s strani ponudnika identitete, lahko ponudnik storitve odgovori uporabniškemu agentu glavnice, tako da postavi svoj lasten varnostni kontekst za glavnico in ji vrne želeni vir.

ECP- (Enhanced Client And Proxy) profil: definira, kako je protokolno sporočilo `<AuthnResponse>` uporabljeno v kombinaciji z vezavo povratno SOAP (PAOS), namenjeno podpori mobilnim napravam, ki uporabljajo WAP-dostop.

Enkratno prijavo dosežemo z uporabo ECP tako, da sledimo naslednjim korakom. Najprej uporabnik prek ECP ustvari HTTP-zahtevo za varovani vir na ponudniku storitve, kjer ponudnik storitve nima vzpostavljenega varnostnega konteksta za ECP in uporabnika. Nato ponudnik storitve izda `<AuthnReqest>` sporočilu na ECP, ki je dostavljeno s strani ECP na pravilnega ponudnika identitete. Za to se uporabi PAOS-vez. ECP pridobi lokacijo končne točke ponudnika identitete za protokol avtentikacijske zahteve, ki podpira želeno vez. ECP nato prenese `<AuthnRequest>` na ponudnika identitete z uporabo modificirane verzije SAML-/ SOAP-vezave skupaj z dovoljenjem, da lahko ponudnik identitete izmenjuje HTTP-sporočila z ECP, preden odgovori SAML-zahtevi. Ko ponudnik identitete identificira uporabnika, izda `<Response>` sporočilo, ki ga ECP dostavi ponudniku storitve z uporabo SAML-/SOAP-vezi. Ob prispelem sporočilu ponudnik storitve dodeli varovani vir uporabniku.

Profil odkritja ponudnika identitete: definira, kako lahko ponudnik storitve odkrije, kateri ponudnik identitete se uporablja znotraj omrežnega strežnika.

Recimo, da primer UK (primer.co.uk) in primer DE (primer.de) spadata pod virtualno organizacijo primer.com. V tem primeru je primer.com skupna domena. Oba, primer UK in primer DE, imata prisotnost na tej domeni. Piškotek skupne domene je varen brskalni piškotek, vezan na skupno domeno. Za vsakega uporabnika ta piškotek shranjuje, katere vse ponudnike identitete je uporabnik obiskal. Po uspešnem dejanju avtentikacije je ponudnik identitete zahteval storitev pisanja piškotka skupne domene. Ta storitev pripne unikaten identifikator ponudnika identitete na piškotek. Ponudnik storitve ob zahtevi za varovani vir zahteva storitev branja piškotka skupne domene, da odkrije, na katerem ponudniku identitete je bil uporabnik nazadnje.

Profil enkratne odjave: definira profil SAML-enkratne odjave. Definira, kako so vezi SOAP, HTTP-preusmeritvene, HTTP POST in

HTTP artefakt uporabljene za doseganje enkratne odjave. Ko je uporabnik enkrat prijavljen s ponudnikom identitete, lahko ta entiteta vzpostavi sejo z uporabnikom, ponavadi to stori s piškotkom ali spremembo URL-zapisa. V nekem kasnejšem trenutku želi uporabnik končati določeno sejo bodisi z določenim udeležencem seje ali kar z vsemi udeleženci v določeni seji, upravljani s strani organa sej.

Prekinitev seje poteka tako, da udeleženec seje sproži enkratno odjavo in konča sejo s pošiljanjem `<LogoutRequest>` sporočila ponudniku identitete, od katerega je dobil avtentikacijsko trditev. Zahteva je lahko poslana direktno prek ponudnika identitete ali indirektno prek uporabniškega agenta. Nato ponudnik identitete uporabi vsebino `<Logout Request>` sporočila, da determinira, katere seje je treba končati. Udeleženec seje nato konča sejo uporabnika, kot je bilo zahtevano, in vrne `<LogoutResponse>` ponudniku identitete. Za konec ponudnik identitete preda še `<LogoutResponse>` originalnemu zahevniku prekinitve seje.

Profil menedžmenta identifikacije imena: definira, kako je protokol za menedžment identifikacije imena lahko uporabljen z vezmi SOAP, HTTP prevsmeritveno, HTTP POST in HTTP artefakt. V scenariju na podlagi profila menedžmenta identifikacije imena je ponudnik identitete izmenjal neke vrste identifikator za glavnico s ponudnikom storitve, ki jim omogoča, da delijo skupen identifikator za nekaj časa.

Deluje tako, da ponudnik identitete ali storitve sproži profil s pošiljanjem `<ManageNameIDRequest>` sporočila drugemu ponudniku, ki ga želi obvestiti o spremembi. Nato ponudnik, ki se je odzval, izda sporočilo `<ManageNameIDResponse>` prvotnemu ponudniku zahteve.

3.4 Varnost in zasebnost SAML

Glavni namen SAML-ja je varnostna izmenjava informacij med strankami in zasebnost uporabnikovih podatkov. Uporabnikom mora biti zagotovljeno, da njihovi osebni podatki in atributi, ki jih zagotavljajo, niso ne videni ne uporabljeni s strani neavtorizirane entitete iz katerega koli razloga.

Varnost

SAML uporablja niz varnostnih mehanizmov za preprečevanje, detekcijo in ukrepanje proti morebitnim napadom. Prvi varnostni ukrep je, da si zaupanja vredne stranke in trditvene stranke zaupajo v medsebojno sodelovanje, ki tipično temelji na PKI (Public Key Infrastructure). Četudi uporaba infrastrukture javnega ključa ni predpisana s strani SAML-protokola, je zelo priporočljiva. SAML-specifikacija priporoča in v nekaterih primerih zahteva različne varnostne mehanizme, kot so TLS 1.0+ za varnost na ravni transporta ter XML-podpis in XML-enkripicija za varnost na ravni sporočil. Zahteve so pogosto oblikovane v smislu avtentikacije, celovitosti in zaupnosti. Izbiro varnostnega mehanizma prepušča izvajalcem in uporabnikom.

Zasebnost

SAML vključuje možnost, da podaja izjave o atributih in avtorizacijah avtenticiranih entitet. Obstaja zelo veliko pogostih situacij, v katerih ena ali več strani želi obdržati dostopen in hkrati omejen dostop do informacij, ki se prenašajo v izjavah. Tipičen tak primer so izjave zdravstvenih ali finančnih atributov. SAML je sestavljen iz niza mehanizmov, ki podpirajo izvedbo glede na specifikacijo in zahteve zasebnosti: [4]

- Organi preverjanja pristnosti in organi preverjanja atributov lahko zagotovijo neko stopnjo delne anonimnosti z uporabo enkratnih identifikatorjev ali ključev. Ta anonimnost je delna zato, ker je su-

bjekt nujno omejen na niz subjektov v odnosu z avtoriteto. Uporabniki, ki resnično skrbijo za svojo anonimnost, morajo skrbeti, da prekrijejo ali preprečijo nenavadne vzorce ali obnašanja, ki bi lahko pripeljala sčasoma do deanonimizacije.

- Mehanizmi za preverjanje pristnosti v SAML-kontekstu so opredeljeni tako, da omogočajo uporabniku, da se avtenticira na ustrezni ravni varovanosti.
- SAML omogoča uporabniku, da zahteva nekatere postopke, da se prepriča, da so vsi postopki v zvezi z njegovimi podatki o profilu in informacijami ujemajo z obsegom pravic in s politiko uporabnika.

3.5 Prednosti SAML pred drugimi varnostnimi protokoli

SAML je bil dobro sprejet predvsem zaradi treh razlogov, in sicer je standardiziran, varen ter zagotavlja odlično uporabniško izkušnjo:

- **Odprte rešitve**

  SAML je bil opredeljen za pravilno delovanje z različnimi spletnimi protokoli, kot so HTTP, HTTPS, FTP, SMTP, SOAP in drugi. Dodatno SAML tudi podpira več XML-okvirjev.

- **Standariziran**

  Standardizirana SAML-oblika je zasnovana tako, da je interoperabilna s katerim koli sistemom, ki je neodvisen od implementacije. To omogoča bolj odprt pristop k arhitekturi in identiteti federacije brez problemov interoperabilnosti, povezanih s pristopi, specifičnimi za prodajalce.

- **Varnost**

  V moderni dobi računalništva je varnost zelo pomembna, ko gre za aplikacije, ki uporabljene za poslovne namene. SAML se upo-

rablja, da se zagotovi enotna točka za preverjanje pristnosti na varnem ponudniku identitete, kar pomeni, da uporabniške poverilnice nikoli ne zapustijo meje požarnega zidu. Nato se uporablja kar SAML za uveljavljanje identitete drugim. To pomeni, da aplikacije ne potrebujejo shranjevati ali sinhronizirati identitet, kar zagotavlja, da obstaja manj mest, kjer bi lahko vdrli v identitete ali jih ukradli. SAML zagotavlja tudi močno plast varnosti z uporabo PKI (Public Key Infrastructure), da zaščiti identitete proti poskusom napadov.

- **Uporabniška izkušnja**

  Verjetno ena največjih prednosti SAML je uporabniška izkušnja, ki jo zagotavlja. SAML ponuja možnost, da uporabniki varno dostopajo do več aplikacij z enim samim sklopom poverilnic, vpisanih le enkrat. To ponuja enkratna prijava (SSO). Z uporabo SAML lahko uporabniki nemoteno dostopajo do različnih aplikacij, kar jim omogoča, da bolj učinkovito in hitreje poslujejo.

Poglavje 4

Shibboleth kot okvir za implementacijo SAML

V tem poglavju bomo pogledali, kako Shibboleth sploh deluje. Torej bo Shibboleth opisan v nasldnjih sekcijah, ki bojo prikazovale, kako ta implementira SAML prek spletnih strežnikov.

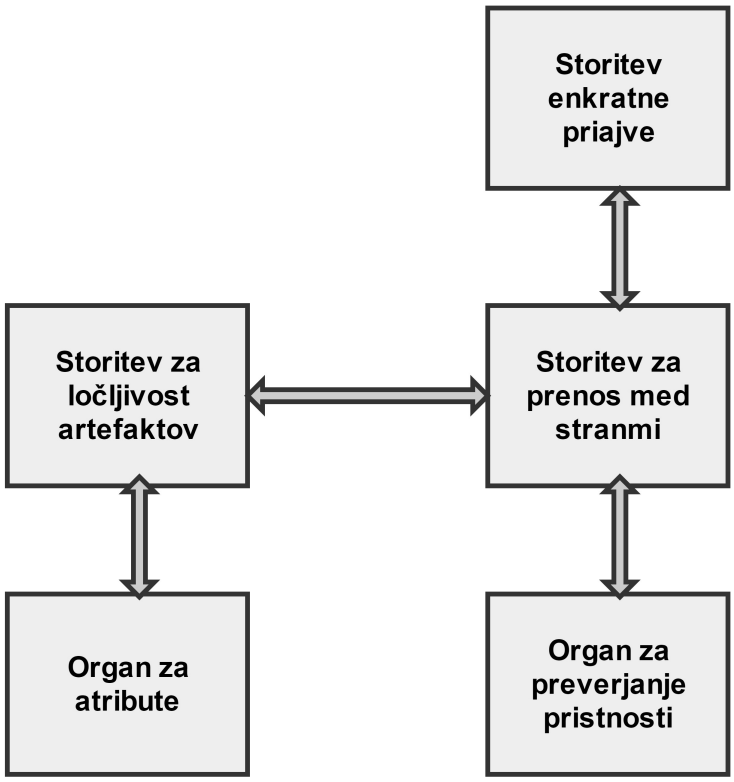
4.1 Shibboleth arhitektura in komponente

Glavne komponente arhitekture Shibboleth sestavljajo ponudnik identitete (IDP), ponudnik storitve (SP) in storitev odkrivanja (DS)(ki je bila prej imenovana WAYF (*Where Are You From*). Poleg teh komponent sta še dva akterja, ki zelo sodelujeta v omrežnem sistemu enkratne prijave, to sta omrežni brskalnik in dostopani vir. Vsi ti elementi sodelujejo med seboj, da se zagotovijo identifikacija, avtentikacija in avtorizacija uporabnika na varen način.

4.1.1 Ponudnik identitete (IDP)

Shibbolethov ponudnik identitete je komponenta, ki je najbolj pomembna za varnostno proceduro. Njena glavna funkcija je, da avtenticira uporabnika skozi obstoječi avtentikacijski sistem, kot je na primer LDAP, in da zagotovi storitev enkratne prijave. Primer prijave je lahko na primer, da neka spletna stran ponuja možnost, da se uporabnik prijavi s poverilnicami Facebook in tako Facebook deluje kot ponudnik identitete. Facebook preverja, ali je uporabnik dejansko pooblaščeni uporabnik, in vrne informacije spletni strani, na katero se prijavljamo, na primer uporabniško ime, elektronski poštni naslov itd. Podobno je, da če stran dovoljuje prijavo s poverilnicami Google ali Twitter, takrat delujeta Google ali Twitter kot ponudnika identitete. Ponudnik identitete je sestavljen iz petih komponent, ki so: organ za preverjanje pristnosti, storitev za prenos med stranmi, storitev ločljivosti artefaktov, storitev enkratne prijave in organ za atribute [7].

Organ za preverjanje pristnosti je sestavljen iz storitev, ki temeljijo na SAML in sprožijo trditve preverjanja pristnosti, za katere se zanimajo entitete, kot je ponudnik storitev. Shibboleth kot sam ne določi, na kakšen način se bodo entitete avtenticirale. Storitev enkratne prijave je izhodiščna točka na nivoju ponudnika identitete (slika 4.1). Na tej ravni sproži postopek preverjanja pristnosti in preusmeri uporabnika proti storitvi za prenos med stranmi, ki je pooblaščena za pridobitev HTTP-odzivov v skladu s profili artefaktov in spletnega brskalnika. Storitev za prenos med stranmi, ki je HTTP-storitev, interoperira z organom za preverjanje pristnosti, da priskrbi zahtevano avtentikacijsko trditev. Tukaj domnevamo, da se uporablja brskalniški/artefakt profil, potem ponudnik identitete priskrbi artefakt ponudniku storitve. Ponudnik storitve nato posreduje artefakt storitvi za ločljivost artefaktov, ki je SAML-protokol, ki bo pomagal ponudniku identitete priskrbeti avtentikacijsko trditev ponudniku storitve. Nato organ za atribute ge-



Slika 4.1: Struktura komponente ponudnika identitete.

nerira zahteve in trditve atributov za identificiranje ter avtenticiranje zahteve, ki jih lahko dobi.

4.1.2 Ponudnik storitve (SP)

Ponudnik storitve je aplikacijska entiteta, ki izvaja spletne storitve ali katere koli spletne vire na podlagi dovoljenja po nekem varnostnem konceptu, ki ga zagotovi SAML-specifikacija. To pomeni, da mora uporabnik pred uporabo želenega vira, ki ga izvaja ponudnik storitve, imeti predhodno veljavne parametre preverjanja pristnosti. [4]

Ponudnik storitve mora vedno imeti unikaten identifikator, ki ga imenujemo entitiyID ali providerID, pri čemer URI ne sme biti daljši od

1024 znakov. Bolj je priporočljiva uporaba HTTPS URL v publikaciji metapodatkov. Shibboleth ponudnik storitve je sestavljen iz naslednjih elementov:

- **Storitev trditev potrošnikov:** je upravljana s strani ponudnika storitve in je HTTP-vir ter končna točka procesa enkratne prijave, ki ima funkcijo procesiranja zahteve HTTP GET ali POST-predložitev profilov, ki želijo vzpostaviti nov varnostni kontekst za entiteto. [4]
- **Zahtevnik atributov:** sčasoma, ko je varnostni kontext ustanovljen, na točki ponudnika storitve implementira povratni kanal izmenjave atributov. [4]
- **Ciljni vir:** je aplikacija ali storitev, do katere želi uporabnik dostopati.

4.1.3 Storitev odkrivanja/WAYF

Storitev odkrivanja, prej imenovana WAYF (*Where Are You From*/Od kod prihajaš), je zelo pomembna komponenta in funkcija Shibboleth, ko gre za reševeanje scenarija, kjer obstaja več kot en ponudnik identitete. V tem primeru storitev odkrivanja pomaga ponudniku storitve identificirati, na kateri ponudnik identitete naj pošlje avtentikacijsko zahtevo. Včasih lahko unikaten uporabnik pripada več ponudnikom identitete in ponudnikom storitve.

4.1.4 Spletni brskalnik in ciljni vir

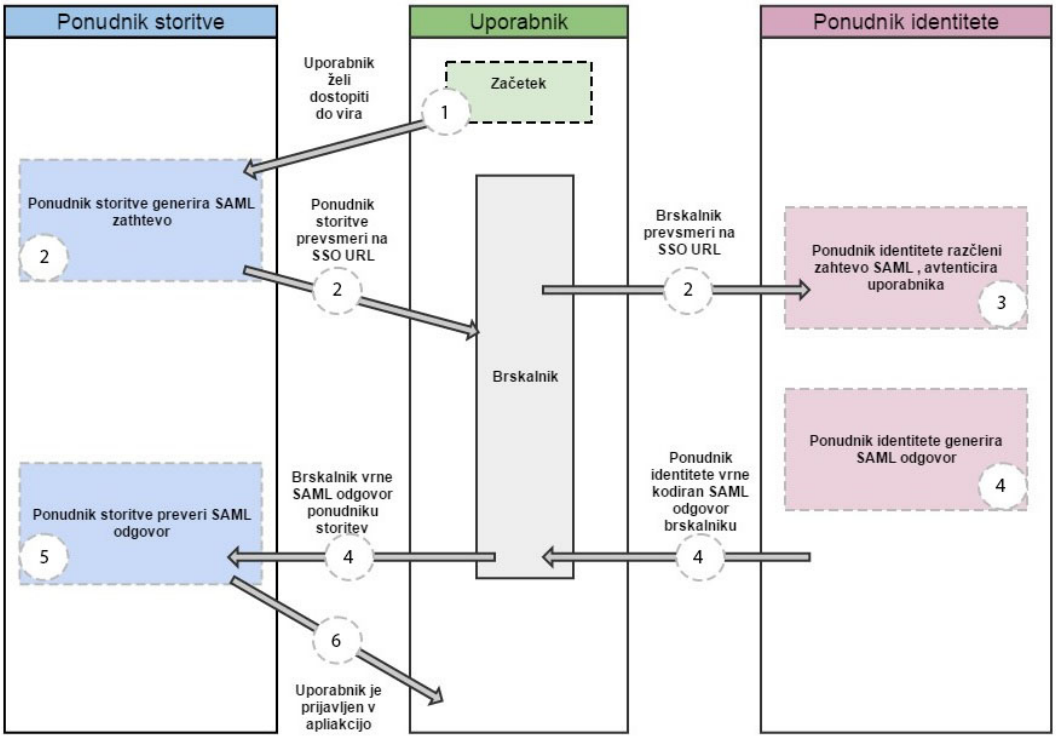
Spletni brskalnik je aplikacija na uporabnikovi strani, ki sproži zahtevo HTTP (s tem ko uporabnik vpiše naslov želene spletne aplikacije) na ponudnika storitve za dostop do zaščitenega vira.

4.2 Koraki enkratne prijave s Shibboleth

V tem delu bomo opisali, kako Shibboelth sploh deluje, kako komponente sodelujejo med seboj, da bi dosegle proces enkratne prijave. Sekvenca (slika 4.2) prikazuje korake in interakcije, ki so zgodijo znotraj tipičnega scenarija enkratne prijave Shibboleth. Glavni akterji so komponente, ki smo jih opisali v zgornji sekciji. Scenarij predvideva, da uporabnik želi dostopati prvič do varovanega vira: [8]

1. **HTTP-zahteva ponudniku storitve:** uporabnik skozi HTTP-zahtevo poskuša dostopati do varovanega vira, ki ga gostuje ponudnik storitve. Upravljalnik vira nato pregleda, ali ima uporabnik že aktivno sejo ali ne. Ko ugotovi, da uporabnik te seje še nima, je uporabnikova zahteva poslana na ponudnika storitve in s tem se začne operacija enkratne prijave.
2. **Zahteva preverjanja pristnosti, ki jo ponudnik storitve izda ponudniku identitete:** ko ponudnik storitve enkrat dobi zahtevo uporabnika, pripravi zahtevo za preverjanje pristnosti, povezano z zahtevo uporabnika, in vse skupaj preusmeri na ponudnika identitete. Če je več različnih ponudnikov identitete, sta zahteva za preverjanje pristnosti in zahteva uporabnika najprej poslani na storitev odkrivanja, da bi ponudnik storitve določil ter izbral ponudnika identitete, ki mu uporabnik pripada, preden je vse skupaj poslano primernemu. Shibboleth zagotavlja dve storitvi za ukvarjanje s tem problemom. To sta centralizirana storitev odkrivanja in vgrajena storitev odkrivanja. Slednja je bolj zaželena, saj ponuja več uporabniške izkušnje.

3. **Identifikacija in avtentikacija uporabnika s strani ponudnika identitete:** ko ponudniku identitete prispe zahteva s strani ponudnika storitve, ta preveri, ali ima uporabnik že veljavno sejo ali ne. Če jo že ima, postopek nadaljujemo v naslednjem koraku.



Slika 4.2: Primer osnovnega poteka avtorizacije Shibboleth

Če seje še ni, pa mora uporabnik priskrbeti svoje parametre za dostop, kot sta na primer uporabniško ime in geslo, šele nato se postopek nadaljuje.

4. **Ponudnik identitete izvede aveteniticijski odziv na ponudnika storitve:** če so uporabniški parametri bili pravilni in je posledično uporabnik bil identificiran, ponudnik identitete generira SAML-odziv ali več artefakt sporočil, ki štejejo kot avtentikacijski odziv in so poslani nazaj z uporabniško zahtevo na ponudnika storitve.

5. **Ponudnik storitve preveri odgovor:** ko zahteva uporabnika in avtentikacijski odziv prispeta od ponudnika identitete na ponudnika storitve, avtentikacijski odziv opravi validacijo in je tako ustvarjena uporabniška seja s strani ponudnika storitve, ki tudi

priskrbi nekaj pomembnih informacij, kot so uporabniški identifikator, ki bo uporabljen s strani varovanega vira. Po tem je uporabnik prevsmerjen na ciljni vir.

6. **Vir vrne vsebino:** ko vir ve, kdo je uporabnik, ki želi dostopati do vsebine, mu priskrbi zahtevano storitev, aplikacijo ali podatke.

4.2.1 Drugi elementi in funkcije, ki sodelujejo pir Shibboleth enkratni prijavi

V tej podsekciji bomo opisali še nekatere pomembne elemente in funkcije, ki tudi pomagajo Shibbolethu da izvrši protokol enkratne prijave in še več o tem , kako Shibboleth deluje.

Uporabniški atributi: Ena zelo dobrih lastnosti uporabe Shibboletha je napredna zmogljivost ponudnika storitve Shibboleth, da enostavno prejema podatke od ponudnika identitete Shibboleth in obratno. Brez teh podatkov, imenovanih uporabniški atributi, uporabnik ne more biti identificiran in posledično avtenticiran. Atributi so lahko elektronski naslovi, telefonske številke, informacije o skupini, v katero uporabnik sodi, njegova funkcija v organizaciji in tako naprej.

Metapodatki Shibboleth: ponudnik identitete in ponudnik storitve komunicirata prek HTTP ter poznata svoje URL-naslove in več, s pomočjo podatkov o podatkih imenovanimi metapodatki. Metapodatki so dokument, ki v podrobnosti opisuje različne aspekte, povezane s ponudnikom identitete ali ponudnikom storitve. Metapodatki ponudnika storitve morajo biti naloženi v ponudniku identitete in metapodatki ponudnika identitete morajo biti prav tako naloženi v ponudniku storitve za omogočanje komunikacije med njima. Metapodatki ponudnika storitve ali metapodatki ponudnika identitete so sestavljeni iz URL-sporočila, ID-entitete kot identifikatorja, kriptografskih informacij o sporočilih ter s človeško berljivim imenom in opisom. [9] [10]

Zvezna enkratna prijava s Shibboleth: zgornji koraki enkratne prijave so zelo podobni večini sistemom enkratne prijave, čeprav je večina teh sistemov opredeljenih za delovanje samo, če sta ponudnik storitve in ponudnik identitete v isti organizaciji. Implementacija enkratne prijave skozi Shibboleth je mogoča tudi, če ponudnik identitete in ponudnik storitve nista v istem omrežju/organizaciji.

Vezi Shibboleth: Opisujejo in definirajo, kako so sporočila poslana od pošiljatelja do prejemnika. Nekateri primeri vezi vsebujejo POST-vez, ki determinira, kako formatirati, imenovati in poslati sporočilo prejemniku prek zahteve HTTP POST. Vez preusmeritve determinira način, kako poslati sporočilo prek URL-ja HTTP-preusmeritvene zahteve.

Profili Shibboleth: na splošno SAML-profili določijo komponente interoperabilnosti, to pomeni, da če različni izdelki podpirajo sklop opredeljenih profilov, lahko sodelujejo med seboj na nekem določenem nivoju. Specifikacija profilov Shibboleth definira niz funkcij, ki se jih lahko opravlja.

Poglavje 5

Postavitev Shibboleth

Postavili bomo enostavni sistem Shibboleth. Postavitev sistema bo narejena znotraj enostavnega ad-hoc omrežja in bo prikazala osnovno uporabo ter obnašanje Shibboletha. Deloval bo tako, da uporabnik skozi HTTPS URL zahteva varovani vir, ki ga gostuje in nadzira ponudnik storitve. Ponudnik storitve sprejme zahtevo, jo obravnava in pošlje ponudniku identitete, ki je zadolžen za avtentikacijo zahtevo za preverjanje identifikacije. Ponudnik identitete sprejme to zahtevo, jo obravnava in pošlje nazaj avtentikacijski odziv do ponudnika storitve z uporabnikovo avtorizacijo za dostop do varovanega vira. Ko ponudnik storitve prejme odziv, omogoči dostop uporabniku do vira.

5.1 Arhitektura sistema

5.1.1 Strojna oprema

Strojna oprema, uporabljena za izvedbo postavitev, je:

- Za gostovanje ponudnika storitve računalnik Intel core i3 s 500 GB HDD in 4 GB RAM spomina.

- Za gostovanje ponudnika identitete računalnik Intel core i5 z 1 TB HDD in 16 GB RAM spomina.

Oba fizična strežnika sta povezana v lokalno omrežje. Vsak strežnik lahko deluje kot uporabniški klient skozi svoj spletni brskalnik ali pa uporabimo zunanji računalnik, ki deluje kot končni uporabnik. Sistem je lahko vzpostavljen tudi s pomočjo virtualizacije z uporabo VMware workstation ali kakšnega drugega virtualizacijskega programa.

5.1.2 Programska oprema

Programska oprema, potrebna za implementacijo, je sestavljena iz več različnih aplikacij, vseh zelo pomembnih za delovanje implementacije. Napaka že ene same programske opreme lahko vpliva na delovanje celotnega sistema. Ta postavitev temelji na dveh operacijskih sistemih Windows, na katerih so nameščene zahtevane aplikacije (slika 5.1).

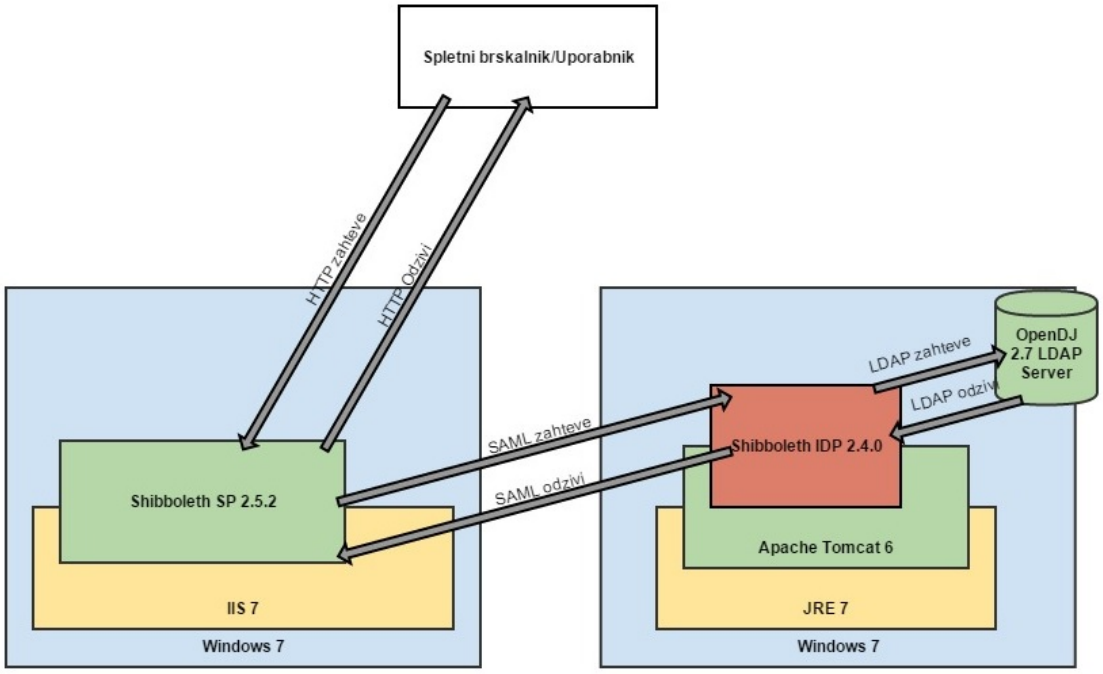
Na strani ponudnika storitve

- Microsoft Windows 7 ultimate edition kot operacijski sistem vseh drugih spodaj navedenih aplikacij za namestitev ponudnika storitve.
- Microsoft Internet Information Services version7 (IIS 7) je fleksibilen, varen in obvladljuv spletni strežnik za gostovanje biločesa na spletu. Njegova prilagodljiva in odprta arhitektura je pripravljena za obravnavo najzahtevnejših nalog. V našem primeru bo uporabljen za gostovanje ponudnika storitve, SSL-certifikata in prve testne aplikacije.
- Shibboleth Service Provider 2.5.2 omogoča za Shibolizirat spletno aplikacijo in posledično varovane vire.

Na strani ponudnika identitete

- Microsoft Windows 7 Ultimate edition kot operacijski sistem vseh drugih aplikacij za namestitev ponudnika identitete.
- Sun Java Client (JRE 7) znan tudi kot Java Runtime je del razvojnega kompleta Java, nabora programskih orodij za razvoj aplikacij Java. Izvajalno okolje določa minimalne zahteve za izvajanje Java aplikacije. Sestavljena je iz Java navideznega stroja, ključnih razredov in podpornih datotek.
- Apache Tomcat 6 je aplikacijski strežnik, ki izvaja Java servlet in renderira spletne strani, ki vsebujejo Java Server Page kodiranja. Je rezultat odprtega sodelovanja razvijalcev in je na voljo na spletni strani Apache. Lahko je uporabljen kot samostojni izdelek, z lastnim notranjem spletnim strežnikom ali pa skupaj z drugimi spletnimi strežniki kot so Apache, Netscape, IIS. Zahteva Java Runtime Enterprise okolje. Uporabili ga bomo, ne kot standardni omrežni strežnik, ampak kot zbiralnik Java servlet z omogočenim SSL.
- Shibboleth Identity Provider 2.4.0 za upravljanje identitet in zagotavljanje avtentikacijskih virov.
- Strežnik OpenDJ 2.7 LDAP kot podatkovna baza za registracijo, shranjevanje uporabnikov in povezavo s ponudnikom identitete za avtentikacijo ter avtorizacijo. OpenDJ obsega odprtokodni imeniški strežnik, orodja strank in LDAP SDK. Razvit je za Java platformo in zagotavlja visoko zmogljivost, visoko razpoložljivost in varno shrambo za identitete, ki jih neko podjetje upravlja. Njegova enostavna namestitev ga uvršča med najpreprostejše in najhitrejše imeniške strežnike za vzpostavljanje in upravljanje. [18]

Slika 5.1 prikazuje arhitekturo programske opreme, ki smo jo uporabili za to implementacijo. Prikazuje interakcijo med različnimi komponentami, ki omogočajo postavitev ponudnika storitve in ponudnika identitete.



Slika 5.1: Struktura postavitve.

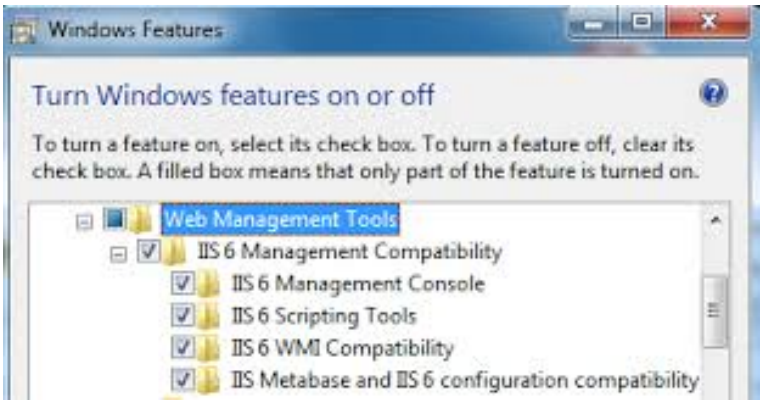
5.2 Postopek postavitve

5.2.1 Namestitev in vzpostavitev ponudnika storitve Shibboleth

Pogoji za implementacijo ponudnika storitve so namestitev in konfiguracija IIS 7 ter generiranje in vezava samopodpisanega SSL-certifikata za enkripcijo prihajajočih ter odhajajočih paketkov.

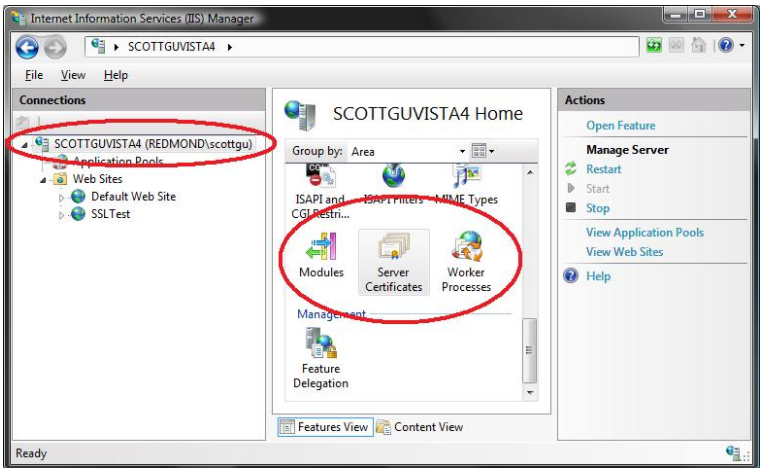
5.2.1.1 Namestitev in konfiguracija IIS 7

Windows 7 ultimate že avtomatsko vsebuje orodja IIS 7, treba jih je le omogočiti in namestiti glede na dokumentacijo Shibboleth SP. Ob namestitvi je treba obkljukati *IIS 6 Managment Compatibility* storitev, ki bo pomagala Shibbolethu uporabljati upravljalne vmesnike in opraviti nekatere avtomatske konfiguracije (slika 5.2). Nato je treba namestiti tudi "ISAPI filters and extensions".



Slika 5.2: IIS-namestitev

Za zavarovanje spletne strani IIS 7 je pomembno generirati in namestiti certifikat SSL x509 ter ga vezati na spletno stran. Za to storiti gremo na IIS-storitev in dvakrat kliknemo na *Server Certificates* opcijo (slika 5.3). Nato kliknemo na *Create Self Signed Certificate* in ga kreiramo ter vežemo na IIS-privzeto stran. Za testiranje, ali certifikat deluje, se povežemo na https://localhost/ v spletnem brskalniku, kjer bi nam moralo pokazati varnostno sporočilo, ki sprašuje za dovoljenje za nadaljevanje.



Slika 5.3: Kreiranje certifikata.

5.2.1.2 Namestitev Shibboleth SP 2.5.2

Najprej prenesemo datoteko shibboleth.msi s spletne strani shibboleth.net, zaženemo installer in nastavimo parametre tako, da obkljukamo *Install ISAPI modules into IIS* ter nastavimo *IIS Script Extension* na .sso. Ko je namestitev zaključena, preverimo, ali je Shibboleth SP integriran z IIS, to storimo tako, da odpremo IIS-upravitelja in preverimo, ali ISAPI-filter nameščen, tako da kliknemo na ime strežnika in odpremo ISAPI-filter zavihek.

Integracija je uspešna, če na ISAPI-filtrih piše: *Name: Shibboleth,Execut:* `C:/opt/shibboleth-sp/lib/shibboleth/isapi-shib.dll|`. Zdaj je Shibboleth nameščen in integriran z IIS. Da dokončno zagotovimo, da je Shibboleth bil nameščen in deluje pravilno, naredimo test tako, da v brskalnik vnesemo `https://localhost/Shibboleth.sso/Status`, da preverimo stanje Shibboleth. Drugi način, s katerim lahko preverimo delovanje, je, da v ukazno vrstico vnesemo:

```
C:/opt/shibboleth-sp/sbin>shibd.exe -check
```

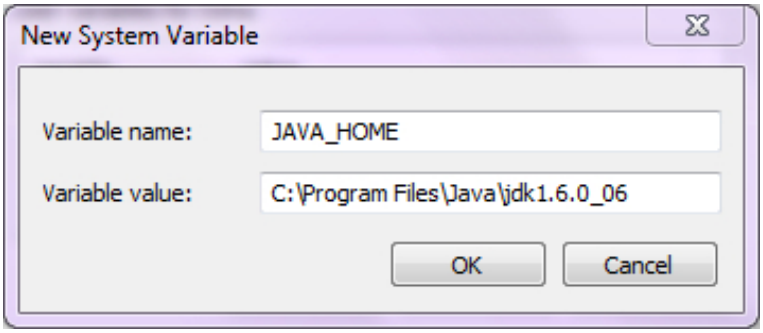
Te ukaze lahko vedno uporabljamo za preverjanje, ali ponudnik storitve pravilno deluje. Ko je test uspešen, pomeni, da ponudnik storitve

pravilno deluje in je pripravljen na konfiguracijo.

5.2.2 Namestitev in vzpostavitev ponudnika identitete Shibboleth

V teji sekciji bomo prikazali namestitev in vzpostavitev ponudnika identitete.

5.2.2.1 Namestitev in konfiguracija JRE 7



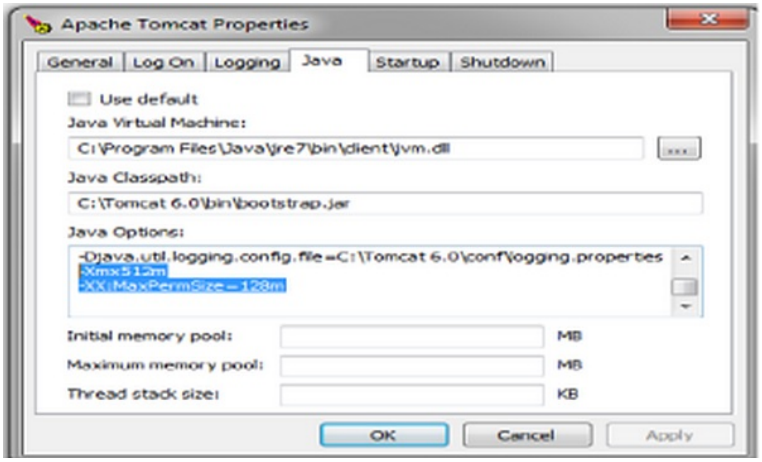
Slika 5.4: Kreiranje spremenljivke okolja

Java Runtime Environment (JRE) potrebujemo za delovanje Tomcata in ponudnika identitete, saj brez tega to ni mogoče. JRE 7 je tudi potreben za namestitev podatkovne baze OpenDJ LDAP. Ko namestimo JRE 7, moramo v Windowsih nastaviti `JAVA_HOME` okolijske spremenljivke. To storimo tako, da gremo na:*Dodatne nastavitve sistema - Sistemske lastnosti - Dodatno - Spremenljivke okolja* in pod uporabniškimi spremenljivkami kliknemo Nova ter ustvarimo `JAVA_HOME` za ime spremenljivke in pot, kjer smo namestili JRE za njeno vrednost (slika 5.4).

5.2.2.2 Namestitev in konfiguracija Apache Tomcat 6

Za namestitev tomcata je treba navesti le pot, kamor bomo tomcat namestili, HTTP/1.1 konektor na 8080, priskrbeti administrativne pravice in pot prej nameščenega JRE. Pravilno namestitev preverimo tako, da v brskalniku dostopimo do `http://localhost:8080` po zagnani Tomcat storitvi. Če deluje, se na strani prikaže domača stran apache tomcat.

Naslednji korak je konfiguracija v zvezi s ponudnikom identitete. Najprej moramo potrditi knjižnjici Xerces in Xalan, kar storimo tako, da ustvarimo mapo, imenovano *endorsed* v Tomcat 6 direktoriju. Nato kopiramo vse datoteke .jar iz *endorsed* direktorija na lokaciji, kjer smo namestili ponudnika identitete v na novo ustvarjen *endorsed* direktorij Tomcata. Nato moramo nastaviti `JAVA_OPTS` okolijsko spremenljivko v Tomcatu - to storimo tako v tomcat nastavitvah pod Java zavihkom; v Java Options polje vnesemo `-Xmx512m` in `XX:MaxPermSize=128m` (slika 5.5). 512 je vrednost dovoljenih megabitov in 128 največja možna vrednost spomina, dodeljenega za generacijo objektov. Ker je namen



Slika 5.5: Nastavitev java spremenljivke v tomcatu

ponudnika storitve Shibboleth in ponudnika identitete komuniciranje

med seboj, moramo konfigurirati dodaten port, imenovan *Connector* za varno komunikacijo. To bo pomagalo obem ponudnikom pri izboljšavi varnostnih zahtev. Da to storimo, moramo pridobiti knjižnjico `tomcat6-dta-ssl-1.0.0.jar` in jo postaviti v Tomcat lib direktorij. Ko to storimo, moramo modificirati datoteko server.xml, tako da ji dodamo spodnji del kode:

```
<Connector port="8443"
   protocol="org.apache.coyote.http11.Http11Protocol"
   SSLImplementation="edu.internet2.middleware.security.tomcat6
   .DelegateToApplication nJSSEImplementation"
   scheme="https"
   SSLEnabled="true"
   clientAuth="want"
   keystoreFile="C:\IDP\credentials\idp.jks"
   keystorePass="mojegeslo" />
```

S tem bomo Tomcatu povedali, kje je certifikat idp.jks, ki je certifikat, zagotovljen s strani Shibboletha, ki bo pomagal varovati Tomcat z omogočanjem SSL-protokola.

5.2.2.3 Namestitev ponudnika identitete

Ponudnika identitete namestimo tako, da zaženemo datoteko Shibboleth IDP .msi in sledimo navodilom. Ko to storimo, moramo narediti namestitveno strategijo idp.war na Tomcatu. Ta konfiguracija določa, kako bo tomcat namestil spletno aplikacijo idp.war. Tehnika se imenuje *Context Deployment Fragment* in deluje tako, da ustvarimo kratko XML-kodo, imenovano `idp.xml`, ter jo postavimo v mapo `/conf/Catalina/localhost/` v Tomcat direktoriju. Vsebina idp.xml datoteke je:

```
<Context docBase="C:\IDP\war\idp.war"
       privileged="true"
```

```
antiResourceLocking="false"
unpackWAR="false"
swallowOutput="true" />
```

Koda pove Tomcatu, kje dobi idp.war za namestitev, in tudi določi Tomcatu lastnosti spletne aplikacije.

5.2.3 Omrežni dostop

Preden preidemo na konfiguracijo ponudnika storitve in ponudnika identitete, moramo najprej vzpostaviti IP-omrežje z uspešno razrešenim IP-naslovom do lažnega gostiteljskega naslova. Popolna kvalificirana imena domene se imenujejo *full quality domain names* (FQDN). FQDN je ena glavnih Shibboletovih priporočljivih zahtev, predvsem ko gre za določitev identitete.

5.2.3.1 Konfiguracija dostopa do omrežja

Dostop priskrbimo tako, da na enem omrežju vzpostavimo 2 privat IP-naslova in ju konfiguriramo tako, da si lahko med seboj izmenjujeta paketke. Vsakemu strežniku nato določimo lažno gostiteljsko ime in vrata (slika 5.6).

Gostitelj (Strežnik)	OS	IP naslov	Popolno kvalificirano ime domene	http vrata	https vrata
Strežnik Pondunika storitve	Windows 7 Ultimate	192.168.0.2	sp.serv.com	80	443
Strežnik Ponudnika identitete	Windows 7 Ultimate	192.168.0.3	idp.co.uk	8080	8443

Slika 5.6: Določene omrežne vrednosti za FQDN.

Da ustvarimo popolno kvalificirano gostiteljsko ime, gremo v mapo `C:\Windows/System32/drivers/etc` pri obeh strežnikih in spremenimo datoteko host, tako da ji dodamo naslove želenih gostujočijh imen. V našem primeru:

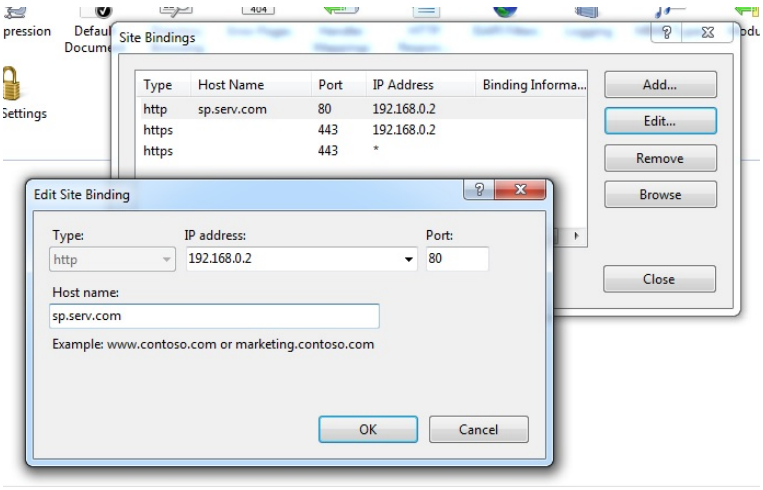
```
192.168.0.2    sp.serv.com
192.168.0.3    idp.co.uk
```

Ko opravimo to konfiguracijo, bi moralo biti mogoče pingati iz ene naprave na drugo tako prek IP-ja kot prek gostujočega naslova.

5.2.3.2 Vezava IP-naslovov na IIS in Tomcat

IIS in Tomcat sta oba po privzetih nastavitvah vezana na IP-naslov 127.0.0.1, ki ustreza gostujočemu naslovu *localhost*. V našem primeru moramo oba vezati na IP in gostujoči naslov, ki smo ju definirali v prejšnjem koraku.

- **Vezava IIS:** Za IIS to storimo tako, da v IIS upravitelju kliknemo na `Bindings>Add`, da vnesemo nov vnos za vez, ki bo primerna našemu IP-naslovu, gostiteljskem naslovu in vratom. (Slika5.7)



Slika 5.7: Vezanje IP-naslova in gostujočega naslova v IIS.

Rezultat zgornje vezave je, da je privzeta stran IIS dostopna prek 192.168.0.2 ali pa prek sp.serv.com, tako s ponudnika storitve kot s ponudnika identitete.

- **Vezava Tomcat** Da enako kot zgoraj storimo tudi v Tomcatu, gremo v `C:/Tomcat 6.0/conf` in spremenimo datoteko server.xml, tako da ji dodamo naslednjo kodo:

```
<Connector port="8080"
        protocol="HTTP/1.1"
        address="192.168.0.3"
        hostname="idp.co.uk"
        connectionTimeout="20000"
        redirectPort="8443"/>
```

Rezultat te vezi Tomcate bo, da bo privzeta stran Tomcat dostopna prek 192.168.0.3 in idp.co.uk tako s ponudnika storitve kot s ponudnika identitete.

5.2.4 Nadaljnja konfiguracija ponudnika storitve

Kakšna konfiguracija je potreba, je odvisno od specifikacij projekta. Konfiguracija je sestavljena iz spreminjanja privzetih vrednosti, ki so bile avtomatično nastavljene ob namestitvi. Za razne napake, ki se ob konfiguraciji pojavljajo, je najbolje pogledati v datoteke log. Konfiguracije, ki smo jih za ta projekt opravili, so naslednje:

- Modificiranje primarne konfiguracije z zamenjavo privzete identitete ponudnika storitve z `https://sp.serv.com/shibboleth`, modifikacija in prilagajanje identitete ponudnika storitve, definiranje predloge sporočil napak in navedbe pravilnega imena gostitelja.
- Omogočiti IIS-spletnemu strežniku, da omogoči ISAPI-filtre s poselitvijo ISAPI-elementov z *sites* elementi, ki odražajo konfigura-

cijo gostitelja, in izkoristiti *RequestMapper* element za uporabo nastavitev vsebine.

- Omogočiti komunikacijo med ponudnikom storitve in ponudnikom identitete z informiranjem ponudnika storitve o lokaciji metapodatkov ponudnika identitete. Metapodatki so potrebni, da si strežnika med seboj zaupata, zato morata oba vedeti, kje so njuni metapodatki. Ta funkcija je nastavljena v `Metadataprovider` elementu.

```
<MetadataProvider type="XML"
file="C:\opt\shibboleth-sp\etc\shibboleth\idp-metadata.xml"/>
```

Zgornja datoteka `idp-metadata.xml` je kopirana iz metadata direktorija ponudnika identitete, da definira in nastavi metapodatke ponudnika identitete, ki so naloženi s strani ponudnika storitve skozi lokalno pot, ki jo nakažemo v zgornjem parametru. To je zato, ker je ta primer postavitve narejen v lokalnem okolju, drugače bi pridobili metapodatke s HTTP-naslova.

5.2.5 Nadaljnja konfiguracija ponudnika identitete

V tem koraku bomo skonfigurirali ponudnika identitete. Konfiguracija se izvaja glede na naslednje korake:

- Najprej moramo zagotoviti, da ponudnik identitete pozna in komunicira s ponudnikom storitve. To storimo tako, kot smo storili pri ponudniku storitve, in sicer, da nakažemo, kje je datoteka z metapodatki ponudnika storitve.

  To je v `C:/IDP/conf/relying-party.xml` datoteki, kjer spremenimo naslednji del kode:

  ```
  <MetadataProvider
  id="ShibbolethMetadata"
  ```

```
xsi:type="metadata:FilesystemMetadataProvider"
metadataFile="C:\IDP\metadata\some-metadata.xml"
maxRefreshDelay="P1D"/>
```

Zgoraj navedena `some-metadata.xml` datoteka je pridobljena tako, da v brskalnik vnesemo naslov:

`https://sp.serv.com/Shibboleth.sso/Metadata`

- Definiramo, katere atribute lahko ponudnik storitve izdaja, tako da jih odkomentiramo v datoteki

  `C:/IDP/conf/attribute-resolver.xml`

5.2.5.1 Kreacija uporabniškega avtentikacijskega sistema

Shibboleth kot sam ne avtenticira uporabnikov, ampak je vezan in uporablja avtentikacijsko metodo ali sistem. Shibboleth podpira več avtentikacijskih metod:

- *External Auth*: Handler delegira drugi avtentikacijski sistem za opravljanje avtorizacije in identifikacije.
- *Username/Password*: poda uporabniku avtentikacijsko stran in nato preveri uporabniško ime ter geslo s tistima, ki sta na LDAP-strežniku.
- *IP Adress*: preveri IP uporabnika glede na obseg danih IP-jev za identifikacijo uporabnika.
- *Previous Session*: se uporablja, ko je uporabljena uporabnikova obstoječa seja kot avtentikacijski mehanizem.

V našem primeru bomo uporabili metodo prijave Username/Password z uporabo LDAP JAAS. Prijave rokovalca konfiguriramo v datoteki `handler.xml` na naslednji način:

```
<ph:LoginHandler xsy:type="ph:UsernamePassword"
jaasConfigurationLocation="file:C:/IDP/conf/login.cong">
```

```
<ph:AuthenticationMethod>urn:oasis:names:tc:SAML:2.0:
ac:classes:PasswordProtectedTransport</ph:AuthenticationMethod>
</ph:LoginHandler>
```

5.2.5.2 Vzpostavitev LDAP-strežnika

LDAP-strežnik, ki ga bomo v tej implementaciji uporabili, je OpenDj, verzija 2.7, ki podpira LDAP3-protokol in integrira LDAP-brskalniški klient ter vnose za hitro inicialno konfiguracijo. Ko opravimo namestitev, sistem avtomatično vnese 20 uporabnikov, oziroma kolikor jih določimo v namestitvenem programu. Teh 20 uporabnikov ustreza ustvarjeni predlogi dvajsetih uporabnikov, ki so lahko modificirani in spremenjeni v realne uporabnike. Najpomembnejše nastavitve med namestitvijo v zvezi s povezavo s ponudnikom identitete so:

- **Fully Qualified Hostname:** idp.com
- **Base DN:** dc=idp,dc=com
- **Password:** geslo123
- **Ldapurl:** localhost
- **Port:** 389

Če se vsebina `login.config` ne ujema z zgornjimi nastavitvami, povezava z LDAP-strežnikom ali avtorizacija ne bo uspela.

5.2.5.3 Konfiguracija prijave in preverjanja pristnosti

Ko je postavitev LDAP-strežnika uspešna, je naslednji korak vzpostavitve avtentikacijskega sistema, in sicer tako, da dodamo naslednjo kodo, katere podatki se moraji ujemati s tistimi, ki smo jih uporabili pri namestitvi LDAP v `login.config` datoteko:

```
ShibUserPassAuth {
   edu.vt.middleware.ldap.jaas.LdapLoginModule required
      ldapUrl="ldap://localhost:389"
      baseDn="dc=idp,dc=com"
      ssl="true"
      userField="uid"
      subtreeSearch="true"
     ServiceCredential="geslo123"
   ;
};
</ph:LoginHandler>
```

Zdaj moramo nastaviti še mehanizem, prek katerega se ponudnik identitete poveže na LDAP-strežnik. To storimo z vnosom naslednje kode v `attribute-resolver.xml` datoteko:

```
<resolver:DataConnector xsi:type="LDAPDirectory"
                xmlns="urn:mace:shibboleth:2.0:resolver:dc"
                       id="myLDAP"
                       ldapURL="ldap://localhost:389"
                       baseDN="dc=idp,dc=com"
                       principalCredential="geslo123">
    <FilterTemplate>
        <![CDATA[
            (uid=${requestContext.principalName})
        ]]>
    </FilterTemplate>
</resolver:DataConnector>
```

5.2.6 Testiranje enkratne prijave Shibboleth

Testiranje deluje tako, da ko uporabnik želi dostopati do vira, na primer `sp.serv.com/index.html`, ki je bil ustvarjen in gostovan v mapi IIS-strežnika `C:\inetpub\wwwroot\secure`. Test je opravljen z uporabo enega od kreiranih uporabnikov v OpenDJ. V našem primeru bo to na primer uporabnik Miha z geslom: mojegeslo123.

Postopek testiranja je sestavljen tako, da uporabnik Miha z brskalnikom Mozilla Firefox zahteva varovani vir (index.html), ki ga gostuje ponudnik storitve. Ponudnik storitve preusmeri uporabnikovo zahtevo na ponudnika identitete, da ta preveri, ali obstaja že obstoječa seja z atributi tega uporabnika. Če obstaja, dobi Miha pravice do želenega vira, če pa ne, potem ponudnik identitete pošlje avtentikacijsko stran ponudniku storitve, da jo ta pošlje Mihi. Miha vnese svoje poverilnice. Ponudnik storitve sprejme te poverilnice in jih preusmeri nazaj do ponudnika identitete. Ponudnik identitete preveri skozi LDAP-podatkovno bazo, da vidi, ali je uporabnik s temi podatki že registriran. Če ni, ga zavrne, če je, ponudnik identitete vrne pozitiven odgovor ponudniku storitve. Ponudnik storitve preveri podpis ponudnika identitete s pomočjo metapodatkov, da preveri, ali je ponudnik identitete verodostojen. Če je, potem Miha pridobi dostop do želenega vira. Za odjavo in končanje seje mora Miha v našem primeru zapreti brskalnik.

Poglavje 6

Zaključek

Delo je bilo sestavljeno iz dveh delov, ki sta pripomogla do končnega izdelka, in sicer osnovne postavitve protokola Shibboleth.

V prvem, teoretičnem delu diplomske naloge smo predelali protokole in storitve, ki so potrebni za implementacijo ter razumevanje protokola Shibboleth. Najprej smo obdelali princip enkratne prijave, ki je glavna lastnost implementacije Shibboleth. Enkratna prijava je dandanes zelo zaželena storitev, saj zelo razbremeni uporabnike pri prijavah na razne aplikacije in storitve, ki jih federacija ponuja. Nato smo obdelali protokol, na katerem Shibboleth sloni, in sicer SAML-protokol, natančneje njegovo osnovno delovanje, arhitekturo, komponente, iz katerih je sestavljen, ter opis varnosti, ki jih ponuja. V zadnjem, teoretičnem delu smo se osredotočili na to, kako Shibboleth sploh deluje. Predvsem smo predstavili njegovo delovanje s SAML-protokolom, opisali njegove pomembne komponente, kot so ponudnik storitve in ponudnik identitete, ter seveda korake, ki potekajo pri prijavi uporabnika na želeni varovani vir.

V drugem delu diplomskega dela smo se lotili postavitve avtentikacije Shibboleth, ki temelji na Windows sistemih v lokalnem omrežju. Cilj postavitve je bil, da uporabnik pridobi dostop do vira s pomočjo enkra-

tne prijave. To smo storili z namestitvijo sistema Shibboleth, se pravi tako ponudnika storitve kot ponudnika identitete, ki delujeta s pomočjo LDAP-podatkovnega strežnika, na katerem so shranjene uporabniške informacije o prijavi. Rezultat postavitve je bil uspešen, ampak je tu še veliko možnosti za izboljšave. Za začetek bi lahko na primer korenito izboljšali varnost postavitve in vgradili Shibboleth še v druge aplikacije ter storitve, ki imajo svojo samostojno avtentikacijo, in se rešili nepotrebnih množičnih avtentikacij ter posledično razbremenili uporabnike. Kot drugo bi lahko Shibboleth postavili na spletni strežnik, da bi bil dostopen tudi zunaj lokalnega omrežja.

Če vse skupaj povzamemo, je postavitev protokola Shibboleth priporočljiva za vse institucije, ki uporabljajo večje število aplikacij in storitev, do katerih morajo njihovi uporabniki dostopati prek neke vrste avtentikacije, naj bo to z uporabniškim imenom in geslom, NTF-kartico ter drugimi.

Literatura

[1] Shibboleth Wiki.(2014) Shibboleth. [Online]. Dosegljivo: https://wiki.shibboleth.net/

[2] Shibboleth.(2014) about. [Online]. Dosegljivo: https://shibboleth.net/about/

[3] web.mit.edu (2014) Kerberos. [Online]. Dosegljivo: http://web.mit.edu/kerberos/

[4] Oasis-open.(2008) Saml Tech Overview [Online]. Dosegljivo: https://www.oasis-open.org/committees/download.php/27819/sstc-saml-tech-overview-2.0-cd-02.pdf

[5] Cantor,S. in Erdos,M.(2002) Shibboleth-Architecture DRAFT v05 [Online]. Dosegljivo: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=4C259DD965-8464A46313CAD74CF904CC?doi=10.1.1.119.7115rep=rep1type=pdf

[6] Shibboleth Wiki (2014) Installation and Configuration [Online]. Dosegljivo: https://wiki.shibboleth.net/confluence/display/SHIB2/Installation+and+C-onfiguration

[7] Shibboleth Wiki(2005) Shibboleth arhitecture [Online]. Dosegljivo: https://wiki.shibboleth.net/confluence/download/attachments/2162702/i-nternet2-mace-shibboleth-arch-protocols-200509.pdf

[8] Shibboleth.(2014) Basics. [Online]. Dosegljivo: https://shibboleth.net/about/basic.html

[9] saml.xml.org(2014) SAML 2.0. [Online]. Dosegljivo: `http://saml.xml.org/saml-specifications`

[10] Shibboleth.(2014) Intermediate [Online]. Dosegljivo: https://shibboleth.net/about/intermediate.html

[11] Microsoft.(2014) Integrated Windows Authentication [Online]. Dosegljivo: http://www.microsoft.com/technet/prodtechnol/WindowsServer 2003/Library/IIS/523ae943-5e6a-4200-9103-9808baa00157.mspx?mfr=true

[12] OpenIDexplained. (2014) OpenID [Online]. Dosegljivo: http://openidexplained.com/

[13] A. Kumar (2014) MODELING AND ANALYZING WEB PROTOCOLS FOR TRUST AND SECRECY [Online]. Dosegljivo: http://alloy.mit.edu/alloy/papers/kumar-thesis.pdf

[14] A. Pashalidis, C. J. Mitchell (2014) Single Sign-On using Trusted Platforms [Online] Dosegljivo: https://repository.royalholloway.ac.uk/file/8d20369c-e2bd-d23b-d320-249b7ec0ff72/9/ssoutp2.pdf

[15] A.Mayer (2013) On the Security of Web Single Sign-On [Online] Dosegljivo: http://www-brs.ub.ruhr-uni-bochum.de/netahtml/HSS/Diss/MayerAndreas/diss.pdf

[16] K. D. Lewis, J. E. Lewis (2009) Web Single Sign-on Authentication using SAML [Online] Dosegljivo: http://arxiv.org/ftp/arxiv/papers/0909/0909.2368.pdf

[17] T. Gert Roessler (2002) Identification and Authentication in Networks enabling Signle Sign-On [Online] Dosegljivo: https://online.tugraz.at/tug$_o nline/voe_m ain2.getvolltext?pCurrPk =$ 38792

[18] Sermersheim, J., Ed. (2006). Lightweight Directory Access Protocol (LDAP): The Protocol. RFC 4511.